Multi-path Multimedia Streaming System: Adaptation and Optimization

Chow Lik Hang

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of
Master of Philosophy
in
Computer Science and Engineering

June, 2003

Abstract

Quality of service (QoS) in delivery of continuous media over the Internet is still relatively poor and inconsistent. Although many streaming applications can tolerate some degree of missing information, significant information losses will degrade and affect an application's quality. One approach to providing QoS for continuous media applications over the Internet is to use the IntServ model for signaling (e.g., RSVP) and resources reservation in all routers along the streaming path. However, this approach suffers from scalability and deployment problems.

In contrast, we investigate the potential benefits of mitigating the QoS guarantee problem through the exploitation of multiple paths, not necessarily independent paths, existing in the network between a set of senders and a receiver of continuous media. One advantage of this approach is that the complexity of QoS provision can be pushed to the network edge which improves the scalability and deployment characteristics. At the same time, it provides a certain level of QoS guarantees. Specifically, we consider pre-recorded continuous media applications such as those using in video-on-demand systems. We use the following metrics to evaluate the performance of multi-path streaming as compared to single-path streaming: (a) data loss rate, (b) conditional error burst length distribution given that there is at least one packet loss, and (c) lag1-autocorrelation of losses.

In our work, we first study the benefits of multi-path streaming over the single path approach by considering the conventional Gilbert model, which

characterizes the bursty error nature of a given path. We then extend the work by considering the more detailed functional Gilbert model wherein the loss characteristics of a path depend on the application's transmission bandwidth. Approaches to obtain the optimal splitting of traffic among the multiple paths are given. Our results show significant benefit of multi-path streaming over single-path streaming under the optimal traffic splitting. And these benefits also exhibit when we use erasure codes for error correction.

In this thesis, we also discuss how one can realize and implement the optimal splitting of traffic among different paths and how these design issues affect the performance metrics of interest. Following the proposed multi-path streaming approach, we build a prototype multi-path streaming system. With this prototype, we carry out experiments to validate the proposed multi-path streaming method. The results of this work can be used in guiding the design of multi-path continuous media systems streaming data over best-effort wide-area networks.

摘要

在現今的網際網路上進行多媒體串流服務時的質量仍然是相對地差劣和不一致。 雖然大部份的多媒體串流程式都可以容忍某程度上的資訊損失，但是重大的資訊損失將影響多媒體服務的質量。 其中一種在網際網路上提供質量保證的方法是使用綜合服務模型爲通訊協定，並且在串流途徑中的所有路由器內預留資源。 然而，這種方法在大規模推行上面對很大的困難。

相反，我們透過利用存在於多媒體發放者與接收者間的多個通道來提供某一水平的質量保證。使用這種方法的好處是複雜的運算只需要在網路邊緣上進行，而不需要改動網路核心。 這提高在大規模推行上的方便程度。 我們考慮被預先錄製的多媒體串流系統及使用 (一) 資料損失率、 (二) 有條件連續錯誤長度分佈、 和 (三) 單差距自相關數 來比較多重道流與單一道流的質量分別。

我們首先使用吉爾伯特模型來描繪一個道流的連續錯誤特質，並且以此分析多重道流比單一道流較優的地方。 然後，我們考慮比較詳細的函數型吉爾伯特模型。 在這模型中，一個道流的錯誤特質會被應用程式在該道流上使用的傳輸帶寬影響。 我們提出了在多重道流上尋找最優流量分配的方法。 我們的結果顯示，在多重道流上使用最優流量分配時會比單一道流有明顯優勝的地方。 這些好處無論在除錯誤代碼使用與否時都能表現出來。

在這份論文，我們並且討論如可能實現最優流量分配的方法並且談及這些設計方式對質量的影響。 依據我們提出的多重道流串流方案，我們建立了一個原型串流系統。 在這個原型系統上，我們執行了一些實驗以確認這個多重道流方法的成效。 我們希望這工作的結果可幫助網路多重道流串流系統的設計。

Acknowledgments

I wish to express my gratitude to my supervisor Dr. John C.S Lui for giving continuous support and guidance in research. I would like to thank Dr. Leana Golubchik, Dr. G. Franceschinis, and Dr. C. Anglano for giving many helpful advice. I would also like to thank Mr. Adam Lee and Mr. Tak Fu Tung for their support in CSIM and NS-2 simulations. I would like to thank our system administrators Mr. Angus Siu and Mr. Tereance Wong for providing plenty of computer resource and support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Multimedia streaming, especially video streaming, over the current Internet still creates an unpleasant and unsatisfactory experiences. In today's Internet, which is based on the best-effort service model, there is no QoS guarantees for streaming sessions between the servers and the clients. Therefore, packet losses are normal phenomena and theses result in degradation on the viewing quality. In the worst case, streaming sessions may abort unexpectedly. Several approaches such as IntServ model and adaptive media coding have been proposed to alleviate the problem. Although these approaches have their corresponding advantages and shortcomings, solution that is scalable and easily deployable while maintaining quality of service is still not available.

In order to build a scalable and deployable system with consistent quality, we address this problem from a different and unique angle. Our work exploits the fact that there are multiple paths (MP) exist between a set of senders and a receiver. Comparing with the single path (SP) approach, potential benefits on using MP includes a) aggregation of bandwidths so that one can support higher bandwidth applications, b) better loss characteristics which can improve the output quality and improve the eraser code performance, and c) enable adaptation among paths due to network congestion. For example, we can assign different loads on different path at different time instant.

Our approach operates on the application layer, which aims at pushing

the design complexity to the network edge. It improves the scalability and deployment capability while achieving an acceptable level of QoS guarantees.

1.1 Multimedia Streaming Background

We first look at some background on multimedia streaming. Figure 1.1 shows the architecture of a typical streaming system. The sender packetizes the media data and continuously sends the data packets to the receiver. The receiver continuously decodes the media data received and at the same time receives new media data packets. The sending rate of the data packets are often "pace" at the same speed as the media object playback rate. The reason for this "pacing" is to prevent the receiver from getting into the starvation or the overflow problem. Since each packet may experience different amount of delay in the network, therefore, the receiver uses a buffer between the packet receiving and media playback process. This buffer can smooth out the network delay jitter between consecutive receiving packets.

There are many issues in building a good streaming system, for example, data placement in the storage system, media content management, compressing and encoding scheme, etc. In this work, we focus on the network issue only such as to minimize the streaming packet losses in network and reduce the adverse effect caused by these loss packets.

1.2 Streaming over the Internet

A good streaming service requires low loss characteristic and low delay jitter throughout the whole streaming session (which may last for hours). While delay jitter can be absorbed and smoothed out by the buffer at the receiver, packet losses play the main cause on degradation of output quality. Packet losses characteristics not only refers to the average packet loss rate along the

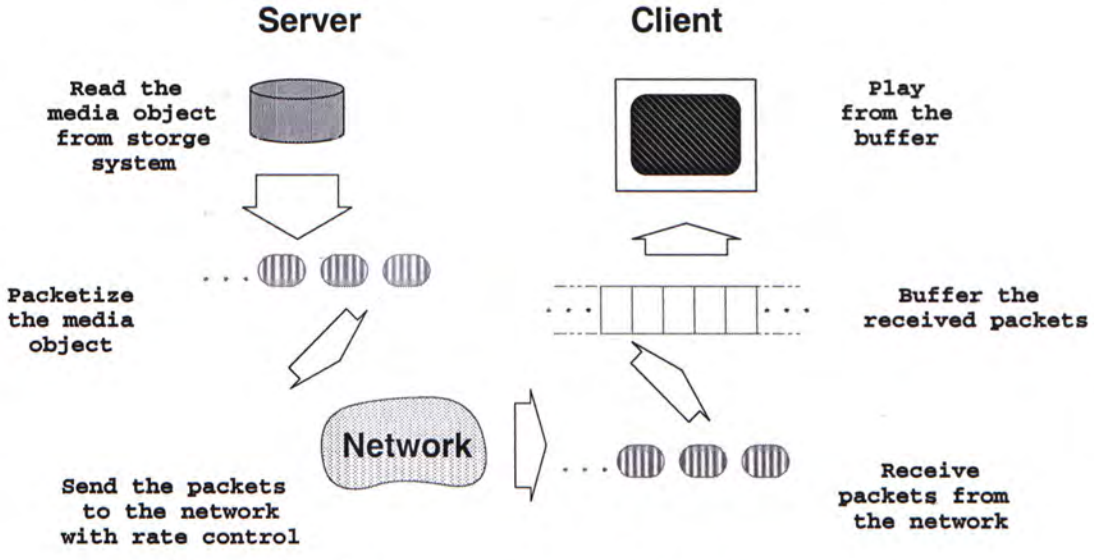


Figure 1.1: Architecture of a typical streaming system.

streaming channel, it also depends on how bursty are the packet losses. In contrast to most file transfer applications, a streaming system normally allows certain degree of missing packets. Many popular media encoding scheme (e.g., MPEG [1]) can tolerate certain level of information loss within a short time period. However, the output quality depends on how bursty are these losses. Section 3.3 presents a more quantitative discussion on how bursty losses affect the video viewing quality.

Insufficient bandwidth definitely worsens the loss characteristic. For example streaming a 1.5Mbps MPEG 1 media via a 56kbps modem connection should results in a high packet loss rate. Although broadband access becomes quite common nowadays, the sharing nature of the best-effort Internet also introduces packet losses.

Figure 1.2 illustrates the abstract idea of an streaming session over the Internet. The server sends the data packets along the a streaming path chosen by the Internet to the receiver. If either router (router 2 in the figure) on the path experiences heavy background cross traffic (indicated by the dotted arrow), it's internal packet queue may get full and packets will be dropped according to the policy configured in the router. In the current best-effort

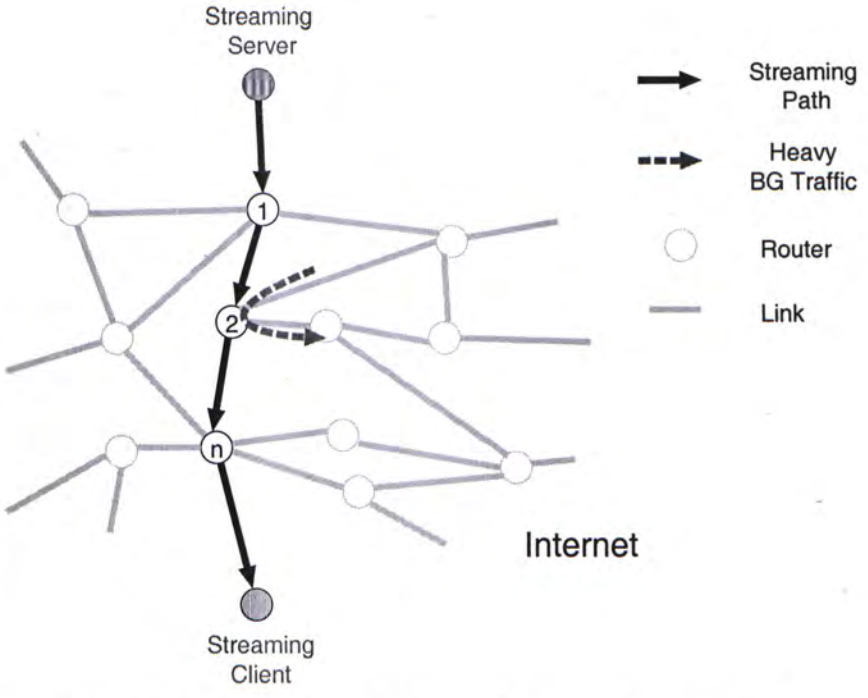


Figure 1.2: Abstract idea on single path streaming over the Internet.

Internet, packets dropped in router 2 belong to the background traffic as well as the streaming session. As part of the streaming information is loss, it results in poorer viewing quality. Also, we say router 2 is the congestion point of the streaming path.

1.3 Traditional Approaches

Various techniques have been proposed aiming at enhancing the quality of streaming application. Note that these approaches are orthogonal and can be applied jointly and independently. Some of these common approaches are:

- Packet Retransmission: The basic idea to deal with packet losses is to retransmit the loss packets. Unlike a file transfer application, each packet in a streaming system has it's timing constraint. Therefor, retransmitted packets that arrived later than it's corresponding playback time are

meaningless to the streaming playback process. The whole packets retransmission process requires at least one round trip time (RTT) delay, which means a precise decision on retransmission triggering is needed. If it is triggered too early, it may introduce duplicate packets at the receiver, which is a waste of network and buffer resources. However, timing constraint may be violated if the retransmission time is triggered too long. To cope with the round trip time (RTT) in the current Internet (which is in the order of milliseconds), the gain and benefit from retransmission is much limited. Another point to notice is that in an overloaded channel, retransmitting packets may worsen the loss characteristic of the channel, which will adversely increase the packet loss rate. The concept of how the increasing workload may worsen a channel is discussed in Chapter 5.

- Adaptive Media Encoding: Newer media encoding schemes support variable bit rate adaption. The streaming application detects the packet loss rate at the receiver. If there are too many packet losses, the server is signaled to encode the media in a lower bit rate (or sends the pre-stored version which was encoded in a lower bit rate). This technique requires accurate network measurement during the streaming sessions. Although it allows a graceful degradation of video output quality, once the stream is adapted to a lower bit rate, the quality of service can still be affected (the video definition is lowered).

- Erasure Channel Coding: Redundant packets can be sent using some erasure-encoding scheme (e.g., Forward Erasure Code (FEC) [2]). If the packet loss rate is not too high, it is then possible that the loss packets can be recovered by the redundant packets. This approach has an adverse effect that sending redundant packets actually increases the application sending rate, which may worsen the channel quality itself. Again, the idea on how the increasing workload may worsen a channel is discussed

in Chapter 5. The amount of redundant packets needed to be accurately adjusted such that it would not overwhelm the network while enable a certain level of packet recovery. Another problem is that if packet losses are too bursty, the packet recovery capability will be significantly reduced. Details on the FEC process is discussed in Chapter 3.

- IntServ Model: This approach has been proposed in the past few years to provide Quality of Service (QoS) guarantee on the Internet. Signaling protocols (e.g., RSVP) are proposed to reserve resources along all the routers on the streaming path. For example, in Figure 1.2, the congested router 2 will not drop the packets belong to the streaming session if it has reserved enough resources (e.g., buffer and transmission bandwidth) for that streaming session. This approach suffers from the deployment and scalability problem. Using Figure 1.2 as an example, if any router on the streaming path does not implemented the RSVP protocol or if any one of them does not have enough resource to fulfill the streaming requirement, the reservation process may fail and the quality may not be guaranteed. As every routers require to work in a per session model, it is extremely not scalable. Newer approach (e.g, DiffServ Model) works in per class basis, yet, it still suffers from the deployment problem.

1.4 Document Road-map

The outline of this thesis is as follow. In Chapter 2, we discuss the related work. Chapter 3 presents the overall idea of our proposed multi-path streaming approach. In Chapter 4, analysis on the benefits of using multiple paths over the single path approach is given by considering the conventional Gilbert model. Extension to the functional Gilbert model is given in Chapter 5. Optimization issues are also presented in this chapter. In Chapter 6, simulations

using Network Simulator(NS) [3] to validate the performance benefits are discussed. Chapter 7 proposes some approaches in quantizing the traffic splitting vector into the sending pattern, which is necessary in real implementation. Our prototype implementation and experiments on this prototype are presented in Chapter 8. In Chapter 9, several design issues and considerations in using the multi-path streaming approach are discussed. Lastly, Chapter 10 concludes our works.

Chapter 2

Related Work

In this chapter we give a brief survey of existing work on this topic, and specifically, we focus on those that either consider loss characteristics or can be deployed over best-effort networks (as these are considerations in our work as well).

Earlier efforts on dealing with losses through the use of multiple independent paths (although at lower layers of the network) include dispersity routing, as proposed by Maxemchuk [4, 5, 6]. The focus in this work was on reducing delay. An important difference in our work is that we focus on streaming applications where the data transmission rate is determined by the application's needs rather than on delivering the data to its destination as fast as possible.

The use of multiple paths in routing data has of course been considered at the network layer, although not generally done in the current Internet. Hence, higher layer mechanisms should be considered. Another set of works on the topic considers higher level mechanisms, but requires some assistance from the lower layers and/or assumes significant knowledge of network topology and/or link capacities and delays (on all links used for data delivery). For instance, in [7], the authors focus on adaptation of delivery rate along the different paths, based on losses observed at the receiver. In contrast, our approach does not rely on specific knowledge of topologies, capacities, delays, etc., and only considers whether a set of paths do or do not share joint points of congestion, as can

be detected at the end-hosts. Moreover, we present optimization schemes for optimizing either the average loss rate or the lag-1 autocorrelation, observed at the receiver.

Recent literature on this topic also includes works on voice-over-IP type applications. For instance, [8, 9] proposes a scheme for real-time audio transmission using multiple independent paths between a single sender and a single receiver, where multiple description coding (MDC) is used in multi-path delivery and a FEC approach is used in single-path delivery. In contrast, we believe that it is important to understand the effects of multi-path delivery on loss characteristics, even without the use of coding techniques. We also note that "live" applications (such as voice-over-IP) have different characteristics than pre-recorded applications (as we are considering here). For instance, one such difference is the need to disperse data in real-time, whereas in our case, we can distribute it to the multiple senders ahead of time; this makes our application-level implementation simpler and possibly more efficient.

In [10], the authors propose a path diversity system using MDC. The system explicitly transmits different encoded packet streams over different paths. Each encoded stream is transmitted through one specific path. In our system, we consider to traffic splitting issue independently of the encoding scheme used. In a later work [11], the authors apply MDC mechanisms over content delivery networks (CDN). In CDN the content is cached and delivered from the closest edge server, so it can potentially reduce the request response time, the probability of packet loss, and the total network resource usage. However, our approach can be deployed over best-effort wide-area networks as well as on CDNs.

Another recent work [12] also considers delivery of pre-recorded video from multiple senders distributed across the network. However, this work focuses on a transport protocol. It also considers optimization algorithms for rate and packet distribution among the paths, but with the objective of only minimizing

the loss rate at the receiver. In [13] FEC techniques are added (as compared to [12]), where distribution algorithms are considered but with the objective of only minimizing the probability of an irrecoverable error. In contrast, due to the nature of the application, we believe that it is important to consider loss characteristics even when the losses cannot be fully recovered. That is, since we are considering delivery of video (which can be displayed even under some losses), it is important to consider other metrics. Hence, in this thesis we consider other metrics as well, i.e., in addition to loss rate we consider burst length distribution as well as lag-1 autocorrelation (all with and without the use of erasure codes). Moreover, in this thesis, we have illustrated the optimization using other loss characteristics, i.e., other than loss rate.

Chapter 3

Our Multi-path Streaming Approach

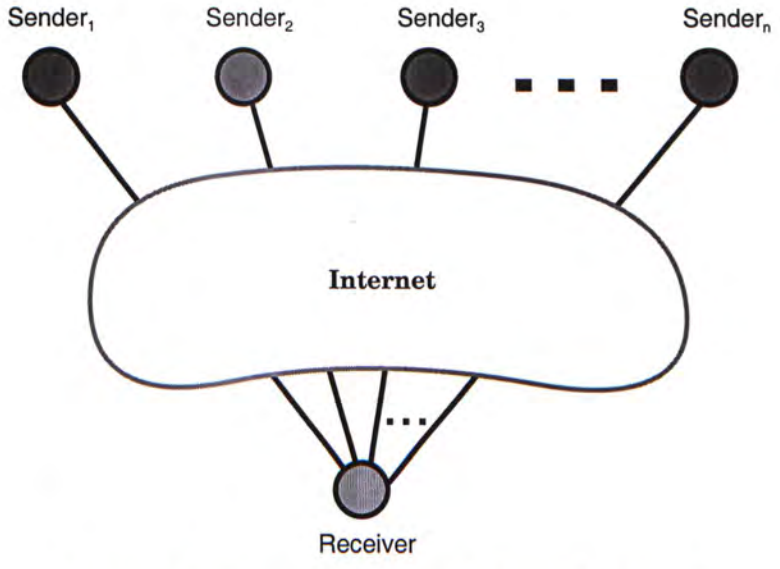


Figure 3.1: Our multiple path streaming model.

There are many possible ways and issues in designing a multi-path streaming system, our scheme operates and focuses on the following:

- Delivery of pre-stored media: Media is pre-stored (as in video-on-demand applications) in contrast to "live" generated (as in video-conferencing applications).

- Multiple senders: We accomplish multiple paths by distributing servers (senders) across the network and stream data to the receiver simultaneously. Each sender delivers a fraction of the streaming data.

- Application layer end-point scheme: Our scheme operates on the application layer, which allows easy deployment. We work at the end-point and treat the network as a "black box". The paths between the senders and receiver are determined by the network-level routing algorithm.

- Network issues only: As mentioned, our work deals with the networking issues. Also, we assume that the media data is fully replicated at all senders already. Other issues like server disk load balancing is outside the scope of this work.

Our system can be depicted in Figure 3.1, where any sender can send any fraction of the media data to the receiver. More specifically, sender i sends fraction α_i of the data to the receiver, where $0 \leq \alpha_i \leq 1$ and $\sum_i \alpha_i = 1$. Sending rate of sender i will be α_i fraction of the media full streaming rate. For example, if a client receives a 1.5 Mbps MPEG1 stream from three senders, according to the network conditions, one can choose: sender 1 sends 50% of the streaming data at 0.75 Mbps, sender 2 sends 30% of the streaming data at 0.4 Mbps and sender 3 sends 20% of the streaming data at 0.3 Mbps. In general, we assume that the setting and the possible adaptation of these fractions (traffic splitting vector) is done by the receiver (based on the perceived quality of data and measurement of the network). The client receives the packets from the senders, reassembles them and plays in the appropriate order.

Erasure coding is adopted in our scheme. We considered a variation of such codes, which we refer to as FEC [2]. Under such scheme, the media file will be divided into groups of data packets such that each group consists of k data packets. Given each group of k data packets, we generate $n > k$ packets. We refer to these n packets as a FEC group. The encoding scheme is such

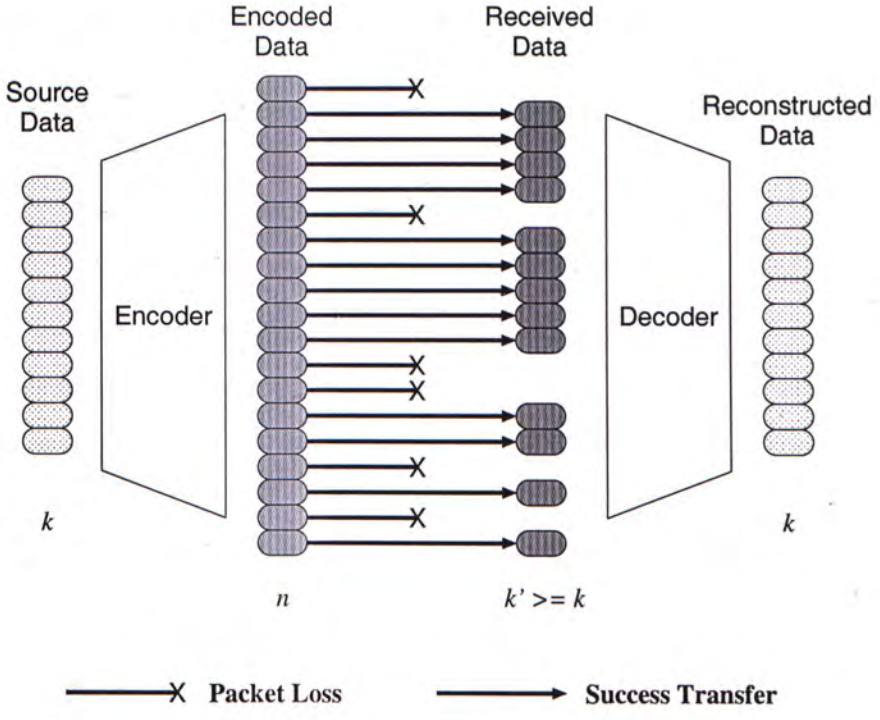


Figure 3.2: Graphical representation of the FEC scheme.

that, if the number of loss packets within a FEC group is less than or equal to $(n-k)$, then we can reconstruct the original k data packets within that FEC group. Figure 3.2 illustrates the idea of such coding scheme. Current FEC technique allows the first k encoded packets remain the same as the original data packets. It brings the flexibility that, even when there are less then k packets received out of n packets in a FEC group, those packets received with number $< k$ still carry meaningful data.

When applying such FEC scheme on multiple paths, we have to consider the value of n and k. The effect of changing n and k will be evaluated in later chapter. Instead of apply FEC on individual path, where packets within individual path form a FEC group, we form FEC groups across all the paths. At a particular time instant, different path carries different loss characteristic. Using such scheme can enhance the FEC correcting ability and can exploit more benefits from multiple paths.

3.1 Potential Benefits

Our multi-path streaming scheme carries the following potential benefits:

1. Increased throughput: bandwidth from multiple paths can be aggregated , so that streaming applications with high bandwidth requirement can be satisfied. It significantly reduces packet loss rate and allows media with higher definition (higher bandwidth requirement) to be serviced.

2. Better loss characteristic:

   - Reduction in correlation between consecutive packet losses: consecutive packets travel along different paths exhibit different loss characteristics. The correlation of consecutive packet losses is reduced in multi-paths streaming and result in less bursty packet losses. This can improve the viewing quality as well as enhance the correcting power of the FEC scheme.

   - Improvement on per-path loss characteristic: loss characteristic for each path is improved as we put less loading on each path. Thus, the overall output quality is enhanced. Detailed discussion will given in Chapter 5.

3. Allow dynamic adaptation among paths: the duration of a streaming session may last last for hours, the characteristic of each channel may vary from time to time during the entire session. Although erasure coding can correct part of the error, using multiple paths enables us to shift part (or all) of the workload from one path to another, so as to avoid the congested path. Another advantage is that under such scheme, we are actually alleviating the network congestion problem. It allows the network to be more efficiently used.

4. Scalable and easy deployable: Application layer end-point scheme is used. Our approach does not require specific support from the network layer so it is easily deployable. All the adaptation processes will be carried out at the senders and receivers only. It shifts the complexity to the end-point, which follows the original design principle of the Internet.

3.2 Performance Metrics

We use the following performance measures to quantify the merits of the different streaming approaches (these are defined more formally below):

1. mean data packets loss rate (with and without FEC),
2. conditional burst length distribution, conditioned on there being at least one error (with and without FEC),
3. lag-1 auto-correlation (with and without FEC).

The first performance measure is an obvious approach to comparing single and multi-path streaming (when losses, rather than throughput, are of importance). The other two performance measures are less obvious; however, we believe that they can significantly affect the quality of the viewed continuous media. To illustrate this point, in the next section we briefly consider a "quality of viewed data" type measure. In subsequent chapters we return to the analysis of the streaming techniques.

3.3 Visual Quality of Data

We first give a brief motivation for considering above given performance metrics, and specifically, for considering burst lengths and correlations between losses. We discuss this in the context of video data. Ideally, one would like

to have a measure of the quality of the viewed video, as a function of loss characteristics. To the best of our knowledge, there is no such widely accepted measure, and often the quality of a video is evaluated using human observers.

However, some metrics have been used in the past, for instance, signal to noise ratio of the resulting video [14]. Hence, we illustrate the effects of bursty losses on the quality of the resulting video (and specifically on the signal to noise ratio) using the following experiment.

3.3.1 Experiment: Effect of Correlated Bursty Losses on Video Quality

In this experiment, we drop 2% of the frames from video $\mathcal{V}$. These 2% losses are introduced in a variety of "patterns", e.g., the dropped frames can be evenly spaced throughout video $\mathcal{V}$, or they can be more bursty. The details of which frames are dropped, given a particular drop pattern as identified by the burst length, are given in the first two columns of Table 3.1. Moreover, in evaluating the quality of the resulting video $\mathcal{V}$, we use a common error concealment scheme to make up for a dropped frame. Specifically, a dropped frame is replaced by the previous frame which is successfully received. For example, frame i replaces frames $i+1, i+2, \cdots, i+k$ if frame i is received successfully and frames $i+1, \cdots, i+k$ are loss.

Error Burst Length	Lost Frames Numbers	PSNR (dB)
1	25+k*50 where k$\in \{0, 1, \cdots, 29\}$	39.107 dB
2	{50,51} + k*100 where k$\in \{0, 1, \cdots, 14\}$	38.015 dB
3	{74,75,76} +k*150 where k$\in \{0, 1, \cdots, 9\}$	31.325 dB
5	{123,124,...,127} +k*200 where k$\in \{0, 1, \cdots, 5\}$	30.433 dB
15	{368,369,...,381,382} +k*750 where k$\in \{0, 1\}$	28.407 dB
30	{736,737,...,764,765}	29.942 dB

Table 3.1: Peak signal-to-noise ratio (PSNR) for various bursty loss patterns.

For each possible frame loss pattern, we measure the quality of the received video by computing the peak signal-to-noise ratio (PSNR) as follows. (Note that, a larger value of PSNR implies a higher quality of the video.) In general, for a video of l frames where each frame consists of $m \times n$ pixels, (each containing an RGB value[1] with each of the three colors represented by 8-bits), the PSNR is calculated using the following expression (in dB):

$$SNR_{peak} = 10 \times \log_{10} \frac{255^2}{\left(\frac{\sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{l} \sum_{c=1}^{3} (P_1(i,j,k,c) - P_2(i,j,k,c))^2}{3 \times m \times n \times l} \right)}.$$

where $P_s(i, j, k, c)$ is the pixel value at coordinate (i, j) of k-th video frame (of stream s, $s = 1, 2$) and color channel c where $c = 1, 2, 3$, for red, green, and blue, respectively. In our experiment, the values of m,n, and l are 352, 240 and 1500, respectively. The source video in this experiment is using MPEG-1 NTSC settings [1] where each frame is 352×240 (with 29.97 frames per second), hence the values of m and n above. Also, we use approximately the first 50 seconds of this video for this experiment, hence the value of l above. Values for P_1 are obtained from the frame sequence resulting after the drop-and-conceal process while values for P_2 are obtained from the original video frames of $\mathcal{V}$.

Table 3.1 gives the PSNR values for the different burst patterns. We can observe that given the same amount of information loss (e.g., 2% in our experiment), the PSNR metric can be significantly lower for the more bursty loss patterns, and hence is the quality of the video. Thus, we believe that burst length distribution and correlations between losses are the right metrics for evaluating the goodness of a streaming approach as they directly reflect on the quality of received video.

[1] Information about the three colors, red, green, and blue.

Chapter 4

Performance Evaluation using Gilbert Model

4.1 Mathematical analysis

In this section, we present our analysis of the single-path and the multi-path streaming approaches. As mentioned earlier, our main focus is on loss characteristics. We first consider these approaches without the use of erasure codes, so as to understand the basic differences between single and multi-path streaming. We then also consider the changes in loss characteristics when an erasure code is added, as this is another approach to dealing with packet losses. Specifically, we consider a variation of such codes, which we refer to as FEC, as defined below. As in [15], we use a two-state Markov chain, known as the Gilbert model, as our model of a path; as in [15] we characterize the path by its bottleneck link. This model, which is defined more formally below, allows for dependence in consecutive packet losses and should be a more accurate representation of the network than an independent loss model.

4.1.1 Model

Let us now state the path model used in this thesis. As in [15], we use a stationary continuous time Gilbert model to characterize the potential correlations between consecutive losses on a path. Under a stationary continuous time Gilbert model, the packet loss process along path k is described by a two state continuous time Markov chain $\{X_k(t)\}$ where $X_k(t) \in \{0, 1\}$. If a packet is transmitted at time t when the state of path k is $X_k(t) = 0$, then no packet loss occurs. On the other hand, the transmitted packet is considered lost if $X_k(t) = 1$. The infinitesimal generator for this Gilbert model of path k is:

$$Q_k = \begin{bmatrix} -\mu_0(k) & \mu_0(k) \\ \mu_1(k) & -\mu_1(k) \end{bmatrix}.$$

The stationary distribution of this Gilbert model is $\boldsymbol{\pi}(k) = [\pi_0(k), \pi_1(k)]$ where $\pi_0(k) = \mu_1(k)/(\mu_0(k) + \mu_1(k))$ and $\pi_1(k) = \mu_0(k)/(\mu_0(k) + \mu_1(k))$. Let $p_{i,j}^{(k)}(\tau)$ be the probability that path k is in state j at time $t + \tau$, given that it was in state i at time t, i.e., $p_{i,j}^{(k)}(\tau) = P(X_k(t + \tau) = j | X_k(t) = i)$. From [16], we have that

$$p_{i,j}^{(k)}(\tau) = \begin{cases} \frac{\mu_1(k)}{\mu_0(k)+\mu_1(k)} \left(1 - e^{-[\mu_0(k)+\mu_1(k)]\tau}\right) & i = 1, j = 0, \\ \frac{\mu_0(k)}{\mu_0(k)+\mu_1(k)} \left(1 - e^{-[\mu_0(k)+\mu_1(k)]\tau}\right) & i = 0, j = 1, \\ \frac{\mu_0(k)+\mu_1(k)e^{-(\mu_0(k)+\mu_1(k))\tau}}{\mu_0(k)+\mu_1(k)} & i = 1, j = 1, \\ \frac{\mu_1(k)+\mu_1(k)e^{-(\mu_0(k)+\mu_1(k))\tau}}{\mu_0(k)+\mu_1(k)} & i = 0, j = 0 \end{cases} \tag{4.1}$$

for all $\tau > 0$.

Throughout the thesis we refer to single path streaming as single path (SP) streaming and multi-path streaming with N paths as multiple path (MP) streaming. Without loss of generality, when paths are homogeneous, we assume that SP streaming always transmits data along path 1. In the evaluation of MP streaming, we assume that the multiple paths have disjoint bottlenecks (or points of congestion) and hence the Gilbert models representing them are independent. Note that, since we represent a path by its bottleneck link,

multiple paths with joint points of congestion could just be represented by a single Gilbert model. Lastly, note that our focus is on a streaming application which generates packets at a constant rate; hence our derivations below are done under this assumption.

4.1.2 Performance Analysis of SP vs. Multi-path Streaming (without FEC)

Let us first derive the average packet loss rate. Unless stated otherwise, below we consider a special case of multi-path streaming, namely dual path, round robin (DPRR) streaming. There are a number of different approaches to distributing data along the multiple paths; here we consider a simple case, i.e., DPRR, wherein each path carries half the application's traffic and the packet transmission is carried out in a round robin manner. That is, odd numbered packets are transmitted along path 1 while even numbered packets are transmitted along path 2. We use this simple scheme for dual path streaming to illustrate the basic performance differences between single and multi-path streaming, so as to gain some basic understanding.

If we assume that the streaming rate does not affect the channel loss characteristics (i.e., the parameters of the Gilbert model), then for the SP case, the average packet loss rate is simply

$$P_{sp}[\text{loss packet}] = \pi_1(1) = \frac{\mu_0(1)}{\mu_0(1)+\mu_1(1)}. \tag{4.2}$$

For the MP case, assume that we have $N \geq 1$ paths and let α_i be the fraction of the application's workload that is sent along path i where $\sum_{i=1}^{N} \alpha_i = 1$. Then the average packet loss rate for the MP case is

$$P_{mp}[\text{loss packet}] = \sum_{i=1}^{N} \alpha_i \pi_1(i) = \sum_{i=1}^{N} \alpha_i \left(\frac{\mu_0(i)}{\mu_0(i)+\mu_1(i)} \right).$$

If these N paths are homogeneous, then we can simplify the above expression

to

$$P_{mp}[\text{loss packet}] = \frac{\mu_0(1)}{\mu_0(1)+\mu_1(1)}. \quad (4.3)$$

Remark: the implication of Equations (4.2) and (4.3) is that if the application's sending rate does not affect the loss characteristics of the path then splitting the data between multiple homogeneous paths does *not* reduce the average packet loss rate, as compared to a single path with the same loss characteristics.

On the other hand, if the application's sending rate can affect the loss characteristics of the path (e.g., sending data with a higher bandwidth may increase the losses), then the average loss rate of the MP approach can be different from that of the SP approach. To illustrate this effect, let λ be the application's mean sending rate and

$$\mu_0(i) = \mathcal{F}(\lambda) \quad (4.4)$$

$$\mu_1(i) = \mathcal{B}(\lambda) \quad (4.5)$$

where $\mathcal{F}$ ($\mathcal{B}$) is a continuous non-decreasing (non-increasing) function of λ. Then, we have the following result.

Theorem 4.1 If the parameters of the Gilbert model are specified by functions $\mathcal{F}$ and $\mathcal{B}$, then the average packet loss rate under the single path streaming approach will be greater than or equal to the average packet loss rate under the multi-path streaming approach wherein these paths have the same Gilbert's parameters.

Proof: It is easy to show that the rate of change of the MP average packet loss rate under the homogeneous Gilbert model is:

$$\begin{aligned}\frac{dP_{mp}[\text{loss packet}]}{d\lambda} &= \frac{d}{d\lambda}\left[\frac{\mathcal{F}(\lambda)}{\mathcal{F}(\lambda)+\mathcal{B}(\lambda)}\right] \\ &= \frac{[\mathcal{F}(\lambda)+\mathcal{B}(\lambda)]\mathcal{F}'(\lambda)-\mathcal{F}(\lambda)[\mathcal{F}'(\lambda)+\mathcal{B}'(\lambda)]}{[\mathcal{F}(\lambda)+\mathcal{B}(\lambda)]^2} \\ &= \frac{\mathcal{B}(\lambda)\mathcal{F}'(\lambda)-\mathcal{F}(\lambda)\mathcal{B}'(\lambda)}{[\mathcal{F}(\lambda)+\mathcal{B}(\lambda)]^2} \geq 0.\end{aligned}$$

That is, a higher sending rate along a path results in a higher loss rate. Since the sending rate along a path in the MP case is less than or equal to the sending rate of the SP case, given that these paths are homogeneous, the resulting average packet loss rate of MP will be less than or equal to that of SP.

Let us now consider the conditional burst length distribution, of both SP and MP cases, conditioned on there being a loss. Let λ_1 be the mean streaming rate (in units of packets per second) along path 1 and $\delta_1 = 1/\lambda_1$ is the time between two consecutively transmitted packets. Then, in the SP case (as also derived in [15] for a voice-over-IP type application), the probability of having a packet error burst of size $m \geq 1$ is:

$$\mathrm{P}_{sp}[\text{error burst} = m] = \begin{cases} \pi_0(1)p_{0,1}^{(1)}(\delta_1)p_{1,0}^{(1)}(\delta_1) & \text{for } m = 1, \\ \pi_0(1)p_{0,1}^{(1)}(\delta_1)\left[p_{1,1}^{(1)}(\delta_1)\right]^{m-1} p_{1,0}^{(1)}(\delta_1) & \text{for } m \geq 2. \end{cases} \tag{4.6}$$

The probability of having a packet error burst of any size is therefore

$$\mathrm{P}_{sp}[\text{error burst}] = \sum_{m=1}^{\infty} \mathrm{P}_{sp}[\text{error burst} = m] = \pi_0(1)p_{0,1}^{(1)}(\delta_1).$$

Moreover, the conditional probability of having a packet error burst of size $m \geq 1$, conditioned on there being a loss, is equal to

$$\begin{aligned} \mathrm{P}_{sp}[\text{error burst of size } m | \text{ error burst}] &= \frac{\mathrm{P}_{sp}[\text{error burst} = m]}{\mathrm{P}_{sp}[\text{error burst}]} \\ &= \left[p_{1,1}^{(1)}(\delta_1)\right]^{m-1} p_{1,0}^{(1)}(\delta_1) \qquad \text{for } m \geq 1. \end{aligned} \tag{4.7}$$

In the MP case, let us consider the special case of DPRR streaming, i.e., $N = 2$. Let λ_2 be the streaming rate (in units of packets per second) along path 1 or path 2. Note that under DPRR, $\lambda_2 = \lambda_1/2$. Then, the time between two consecutively transmitted packets along the same path is $\delta_2 = 1/\lambda_2 = 2\delta_1$. To understand the basic trade off between SP and MP streaming, we also assume that both paths are homogeneous such that they are characterized by a stationary continuous time Gilbert model of the same parameters (i.e.,

$\mu_0(1) = \mu_0(2)$ and $\mu_1(1) = \mu_1(2)$). Given this simplification, the stationary distributions for both paths are the same (i.e., $\pi_0(1) = \pi_0(2)$; $\pi_1(1) = \pi_1(2)$) and we can express all performance measures using the parameters of path 1. Under these assumptions, the probability of having a packet error burst of size $m \geq 1$ is:

$$\begin{aligned} &\mathrm{P}_{dp}[\text{error burst} = m] \\ &\quad = \begin{cases} \pi_0(1)\pi_1(1)p_{0,0}^{(1)}(2\delta_1) & \text{for } m = 1, \\ \pi_0(1)\pi_1(1)\left[p_{1,1}^{(1)}(2\delta_1)\right]^{m-2} p_{0,1}^{(1)}(2\delta_1)p_{1,0}^{(1)}(2\delta_1) & \text{for } m \geq 2. \end{cases} \end{aligned} \tag{4.8}$$

and the probability of having a packet error burst of any size is therefore:

$$\begin{aligned} \mathrm{P}_{dp}[\text{error burst}] &= \sum_{m=1}^{\infty} \mathrm{P}_{dp}[\text{error burst} = m] \\ &= \pi_0(1)\pi_1(1)p_{0,0}^{(1)}(2\delta_1) + \\ &\qquad \sum_{m=2}^{\infty} \pi_0(1)\pi_1(1)[p_{1,1}^{(1)}(2\delta_1)]^{m-2}p_{0,1}^{(1)}(2\delta_1)p_{1,0}^{(1)}(2\delta_1) \\ &= \pi_0(1)\pi_1(1)\left[p_{0,0}^{(1)}(2\delta_1) + \frac{p_{0,1}^{(1)}(2\delta_1)p_{1,0}^{(1)}(2\delta_1)}{1 - p_{1,1}^{(1)}(2\delta_1)}\right] \\ &= \pi_0(1)\pi_1(1)\left[p_{0,0}^{(1)}(2\delta_1) + p_{0,1}^{(1)}(2\delta_1)\right] \\ &= \pi_0(1)\pi_1(1). \end{aligned}$$

Then, the conditional probability of having a packet error burst of size $m \geq 1$, conditioned on there being a packet error, is equal to:

$$\begin{aligned} &\mathrm{P}_{dp}[\text{error burst of size } m | \text{ error burst}] \\ &\quad = \frac{\mathrm{P}_{dp}[\text{error burst} = m]}{\mathrm{P}_{dp}[\text{error burst}]} \\ &\quad = \begin{cases} p_{0,0}^{(1)}(2\delta_1) & \text{for } m = 1, \\ \left[p_{1,1}^{(1)}(2\delta_1)\right]^{m-2} p_{0,1}^{(1)}(2\delta_1)p_{1,0}^{(1)}(2\delta_1) & \text{for } m \geq 2. \end{cases} \end{aligned} \tag{4.9}$$

We can now state the conditions under which the DPRR approach will have a small conditional burst error than the SP approach. Before we present this result, let us present the definition and a basic lemma of stochastic comparison [17].

Definition 4.2 We say that the random variable X is stochastically larger than the random variable Y, written $X \geq_{st} Y$, if $P[X \geq z] \geq P[Y \geq z]$ for all z.

Lemma 4.3 We say that $X \geq_{st} Y$ iff $E[f(X)] \geq E[f(Y)]$ for all increasing functions f.

Now, let $\mathcal{B}_{sp}$ and $\mathcal{B}_{dp}$ be the random variables representing the conditional packet error burst size, given that there is at least one packet error, under the SP and the homogeneous DPRR approaches, respectively. Then, we have the following result.

Theorem 4.4 If $p_{0,1}(2\delta_1)p_{1,0}(2\delta_1) \leq p_{1,1}(\delta_1)p_{1,0}(\delta_1)$, then $\mathcal{B}_{sp} \geq_{st} \mathcal{B}_{dp}$.

Proof: First, note that $p_{1,1}(t)$ is an non-increasing function of t.
If $p_{0,1}(2\delta_1)p_{1,0}(2\delta_1) \leq p_{1,1}(\delta_1)p_{1,0}(\delta_1)$, then from Equations (4.7) and (4.9), we can deduce that for $m \geq 2$,

P_{dp}[error burst of size m| error burst] $\leq \mathrm{P}_{sp}$[error burst of size m| error burst].

Since

$$\sum_{m=1}^{\infty} P_{sp}[\mathcal{B}_{sp} = m] = \sum_{m=1}^{\infty} P_{dp}[\mathcal{B}_{dp} = m] = 1 \qquad \text{and}$$

$$\sum_{m=j}^{\infty} P_{sp}[\mathcal{B}_{sp} = m] \geq \sum_{m=j}^{\infty} P_{dp}[\mathcal{B}_{dp} = m] \qquad \text{for } j \geq 2,$$

we can conclude that $\mathcal{B}_{sp} \geq_{st} \mathcal{B}_{dp}$.

Remark: Note that $\mathcal{B}_{sp} \geq_{st} \mathcal{B}_{dp}$ implies (based on Lemma 4.3) that $E[f(\mathcal{B}_{sp})] \geq E[f(\mathcal{B}_{dp})]$ for all increasing functions f. Therefore, we can conclude that for all moments of $\mathcal{B}_{sp}$ and $\mathcal{B}_{dp}$, we have $E[\mathcal{B}_{sp}^k] \geq E[\mathcal{B}_{dp}^k]$ for $k \geq 1$, where $E[\mathcal{B}_{sp}^k]$ and $E[\mathcal{B}_{dp}^k]$ refer to the k^{th} moments of $\mathcal{B}_{sp}$ and $\mathcal{B}_{dp}$, respectively. The implication of the above theorem is that the homogeneous DPRR approach will have a lower mean conditional burst length than the SP approach, given that the theorem's condition is satisfied.

Let us now consider the lag-1 autocorrelation of packet errors metric. We begin with the SP approach. The lag-1 autocorrelation function $R[X_t X_{t+\delta_1}]$ measures the degree of dependency of consecutive packet errors. For example, a high positive value of $R[X_t X_{t+\delta_1}]$ implies that a lost packet is very likely to be followed by another lost packet. On the other hand, a high negative value of $R[X_t X_{t+\delta_1}]$ implies that a lost packet is likely to be followed by a successful packet arrival. Also, if the statistics of the consecutive packet losses are not correlated[1] , then $R[X_t X_{t+\delta_1}] = 0$.

The lag-1 autocorrelation for the SP approach is

$$R[X_t X_{t+\delta_1}] = \frac{E[(X_t - \overline{X})(X_{t+\delta_1} - \overline{X})]}{E[(X_t - \overline{X})^2]} = \frac{E[X_t X_{t+\delta_1} - \overline{X}^2]}{E[X_t^2 - \overline{X}^2]}.$$

Since $\overline{X} = \pi_1(1) = \mu_0(1)/[\mu_0(1) + \mu_1(2)]$, $E[X_t X_{t+\delta_1}] = \pi_1(1) p_{1,1}^{(1)}(\delta_1)$ and $E[X_t^2] = \pi_1(1) = \mu_0(1)/[\mu_0(1) + \mu_1(2)]$, substituting these expressions into the above equation, gives us

$$\begin{aligned} R[X_t X_{t+\delta_1}] &= \frac{\pi_1(1) p_{1,1}^{(1)}(\delta_1) - \pi_1^2(1)}{\pi_1(1)[1 - \pi_1(1)]} \\ &= \frac{[\mu_0(1) + \mu_1(1)] p_{1,1}^{(1)}(\delta_1) - \mu_0(1)}{\mu_1(1)}. \end{aligned} \tag{4.10}$$

Lemma 4.5 For a high (low) bandwidth streaming application, the lag-1 autocorrelation of the SP streaming approach is positively correlated (tends to zero).

Proof: Note that when $\delta_1 \to 0$, $p_{1,1}^{(1)}(\delta_1) \to 1$, and consequently the lag-1 autocorrelation $R[X_t X_{t+\delta_1}]$ approaches 1. In other words, if the streaming application has a high bandwidth requirement such that the inter-packet spacing tends to zero, then the consecutive packet losses are *"positively"* correlated. On the other hand, when $\delta_1 \to \infty$, $p_{1,1}^{(1)}(\delta_1) \to \mu_0(1)/[\mu_0(1) + \mu_1(1)]$, and consequently the lag-1 autocorrelation $R[X_t X_{t+\delta_1}] \to 0$. This implies that for low

[1]Note that if the lag-1 autocorrelation, $R[X_t X_{t+\delta_1}]$, is equal to 0, it does not necessarily imply that consecutive packet losses are not correlated.

bandwidth streaming applications, wherein the inter-packet spacing is very large, the lag1-autocorrelation tends to zero.

Let us also derive the lag-1 autocorrelation of the homogeneous DPRR approach. The lag-1 autocorrelation in this case is:

$$E[X_t^{(1)}X_{t+\delta_1}^{(2)}] = \frac{E[(X_t^{(1)} - \overline{X^{(1)}})(X_{t+\delta_1}^{(2)} - \overline{X^{(2)}})]}{\sqrt{E[(X_t^{(1)} - \overline{X^{(1)}})^2]E[(X_t^{(2)} - \overline{X^{(2)}})^2]}}. \quad (4.11)$$

Because both paths are homogeneous (i.e., their respective Gilbert models have the same parameters), we can simplify the above expression as:

$$\begin{aligned} E[X_t^{(1)}X_{t+\delta_1}^{(2)}] &= \frac{E[X_t^{(1)}X_{t+\delta_1}^{(2)} - \overline{X^{(1)}}^2]}{E[(X_t^{(1)} - \overline{X^{(1)}})^2]} = \frac{E[X_t^{(1)}X_{t+\delta_1}^{(2)}] - E[\overline{X^{(1)}}^2]}{E[\left(X_t^{(1)}\right)^2] - E[\left(\overline{X^{(1)}}\right)^2]} \\ &= \frac{\left(\frac{\mu_1(1)}{\mu_0(1)+\mu_1(1)}\right)\left(\frac{\mu_1(2)}{\mu_0(2)+\mu_1(2)}\right) - \left(\frac{\mu_1(1)}{\mu_0(1)+\mu_1(1)}\right)^2}{\mu_0(1)\mu_1(1)/\left(\mu_0(1)+\mu_1(1)\right)^2} \\ &= \frac{\left(\frac{\mu_1(1)}{\mu_0(1)+\mu_1(1)}\right)^2 - \left(\frac{\mu_1(1)}{\mu_0(1)+\mu_1(1)}\right)^2}{\mu_0(1)\mu_1(1)/\left(\mu_0(1)+\mu_1(1)\right)^2} \\ &= 0. \end{aligned} \quad (4.12)$$

In fact, we can see that the consecutive packet losses under the homogeneous DPRR application are *"uncorrelated"* since we have assumed independence of the two paths.

4.1.3 Performance Analysis of SP vs. Multi-path Streaming (with FEC)

We have shown that loss characteristics can be improved with multi-path streaming as compared to single path streaming, under conditions and metrics specified above. However, an interesting question that remains is whether there are still benefits to be gained once some form of redundancy is added to the stream. Specifically, we consider the use of an erasure code (as defined below), to which we will refer as FEC in the remainder of the thesis. Hence, in

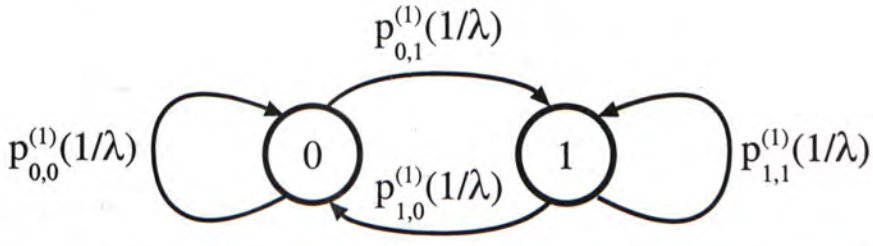


Figure 4.1: An Embedded Markov Chain which describes whether a transmitted packet is loss or not.

this section we focus on the basic understanding of the performance of single path vs. multi-path streaming when FEC is added to the stream.

Since numerous coding schemes exist, we first give the details of the simple FEC scheme considered here. We divide a video file into groups of data packets such that each group consists of k data packets. Given each group of k data packets, we generate $n > k$ packets. We refer to these n packets as a FEC group. The encoding scheme is such that, if the number of lost packets within a FEC group is less than or equal to $(n - k)$, then we can reconstruct the original k data packets within that FEC group.

Let us first derive the average packet loss rate under the SP approach. As before, assume that we use path 1 which is characterized by a Gilbert model, as defined above, with parameters $\mu_0(1)$ and $\mu_1(1)$. The streaming application generates packets at a rate of λ (in unit of packet/sec)[2] . Whenever a packet is transmitted along this path, it may be lost (if the state of the path is "1") or it may arrive successfully at the receiver (if the state of the path is "0"). Figure 4.1 depicts an embedded Markov chain of this path wherein the two consecutive embedded points are $1/\lambda$ units apart. The derivation of transition probabilities of this DTMC is based on Equation (4.1); hence they are a function of the Gilbert model's parameters $\mu_0(1)$ and $\mu_1(1)$ as well as the packet transmission rate λ. The steady state probabilities of this embedded

[2]Note that here, "packets" includes both data packets and packets carrying redundant information.

Markov chain are $\pi_0(1) = \frac{\mu_1(1)}{\mu_0(1)+\mu_1(1)}$ and $\pi_1(1) = \frac{\mu_0(1)}{\mu_0(1)+\mu_1(1)}$.

We are now interested in deriving $P^{(1)}(j,n)$, which is the probability of losing j packet in an n packet transmission. We define

$$P_i^{(1)}(j,n) = \text{Prob}(j,n|\text{initial state of the path is } i) \qquad i \in \{0,1\}$$

as the probability of j lost packet in an n packet transmission, given that the *first* packet was transmitted when the path was in state i (where $i \in \{0,1\}$). We then have:

$$P^{(1)}(j,n) = P_0^{(1)}(j,n)\pi_0(1) + P_1^{(1)}(j,n)\pi_1(1) \quad j = 0,1,\ldots,n. \tag{4.13}$$

We also define for $i \in \{0.1\}$:

$$L_i^{(1)}(j,n) = \text{Prob}(j,n|\text{the initial state of the path is } i \text{ and the final state is } 0)$$
$$H_i^{(1)}(j,n) = \text{Prob}(j,n|\text{the initial state of the path is } i \text{ and the final state is } 1)$$

where $L_i^{(1)}(j,n)$ ($H_i^{(1)}(j,n)$) is the probability that we have j lost packets in an n packet transmission, given that the *first* packet was transmitted when the path was in state i (where $i \in \{0,1\}$) and that the *last* packet was transmitted when the path was in state 0 (state 1). Then we have:

$$P_i^{(1)}(j,n) = L_i^{(1)}(j,n) + H_i^{(1)}(j,n) \quad i \in \{0,1\} \text{ and } j = 0,1,\ldots,n. \tag{4.14}$$

We can also express $L_i^{(1)}(j,n)$ and $H_i^{(1)}(j,n)$ in the following recursive forms: for $j < n$,

$$\begin{aligned} L_i^{(1)}(j,n) &= L_i^{(1)}(j,n-1)(1-p_{0,1}^{(1)}(1/\lambda)) + \\ &\qquad H_i^{(1)}(j,n-1)p_{1,0}^{(1)}(1/\lambda), \end{aligned} \tag{4.15}$$

$$\begin{aligned} H_i^{(1)}(j,n) &= L_i^{(1)}(j-1,n-1)p_{0,1}^{(1)}(1/\lambda) + \\ &\qquad H_i^{(1)}(j-1,n-1)(1-p_{1,0}^{(1)}(1/\lambda)). \end{aligned} \tag{4.16}$$

where we also have the following boundary conditions:

$$L_i^{(1)}(j,m) = 0 \qquad i \in \{0,1\};\ j = 0,1,\ldots,n \text{ and } m \le j \tag{4.17}$$

$$L_0^{(1)}(0,m) = (1 - p_{0,1}^{(1)}(1/\lambda))^{m-1} \qquad \text{for } m = 1,2,\ldots,n \tag{4.18}$$

$$L_1^{(1)}(0,m) = 0 \qquad \text{for } m = 1,2,\ldots,n \tag{4.19}$$

$$H_i^{(1)}(j,m) = 0 \qquad i \in \{0,1\};j = 1,2,\ldots,n \text{ and } m < j \tag{4.20}$$

$$H_i^{(1)}(0,m) = 0 \qquad \text{for } i \in \{0,1\} \text{ and } m = 0,1,\ldots,n \tag{4.21}$$

$$H_0^{(1)}(m,m) = 0 \qquad \text{for } m = 1,2,\ldots,n \tag{4.22}$$

$$H_1^{(1)}(m,m) = (1 - p_{1,0}^{(1)}(1/\lambda))^{m-1} \qquad \text{for } m = 1,2,\ldots,n. \tag{4.23}$$

Remark: To compute the value of $P^{(1)}(j,n)$ in Equation (4.13), we need to compute the values of the four square matrices $\boldsymbol{L}_0^{(1)}, \boldsymbol{L}_1^{(1)}, \boldsymbol{H}_0^{(1)}$, and $\boldsymbol{H}_1^{(1)}$, whose entries can be computed using Equations (4.15) through (4.23). Each of these matrices is of size $(n+1) \times (n+1)$. In other words, computing the values of $P^{(1)}(j,n)$ (for all j) has a computational complexity of $\Theta(4(n+1)^2)$.

Let P_{sp} be the probability of an irrecoverable error within a FEC group. It is equal to

$$\begin{aligned} P_{sp} = \sum_{j=n-k+1}^{n} P^{(1)}(j,n) &= \sum_{j=n-k+1}^{n} \left[P_0^{(1)}(j,n)\pi_0(1) + P_1^{(1)}(j,n)\pi_1(1)\right] \\ &= \sum_{j=n-k+1}^{n} \left[\left(L_0^{(1)}(j,n) + H_0^{(1)}(j,n)\right)\left(\frac{\mu_1(1)}{\mu_0(1)+\mu_1(1)}\right) + \right. \\ &\qquad \left. \left(L_1^{(1)}(j,n) + H_1^{(1)}(j,n)\right)\left(\frac{\mu_0(1)}{\mu_0(1)+\mu_1(1)}\right)\right]. \end{aligned}$$

To derive the average data packet loss rate (with use of FEC) for the SP approach, denoted by $\mathcal{L}_{sp}$, we consider the following two cases, based on the number of lost packets, $j \in \{0,1,\ldots,n\}$, within a FEC group.

Case 1: $j \le n-k$

If j, the number of lost packet within a FEC group, is less than or equal to $n-k$, then all k data packets can be reconstructed at the receiver. Hence, this case does not contribute to *information* loss and $\mathcal{L}_{sp} = 0$.

Case 2: $j > n - k$

In this case, the lost data packets cannot be fully reconstructed and some information will be lost. However, given that there j lost packets within a FEC group, there are a number of different ways to distribute these losses among the n packets of the FEC group. To understand this effect, let us illustrate it using an example. Assume that $n = 5$ and $k = 4$. If $j = 2$, then there are two possible ways to distribute these two lost packets among the packets of the FEC group: (1) the two lost packets are the data packets within the FEC group, or (2) one lost packet is a data packet and the other lost packet corresponds to redundant information in the FEC code. In the first case, we lost 2 data packets out of a 4 data packet transmission. In the second case, we lost 1 data packet out of a 4 data packet transmission. Using the same argument, if $j = 5$, then there is only one way to distribute these five lost packets among packets of the FEC group. That is, all data packets are lost. Therefore, given that there are j lost packets, the number of ways to distribute the j lost packets among the packets of a FEC group is $\mathcal{W} = \mathcal{M} - j + (n-k) + 1$ where $\mathcal{M} = \min\{j, k\}$. Let $\mathcal{L}(j)$ be the average data packet loss rate given that there are j lost packets in a FEC group. Then, we have

$$\begin{aligned}
\mathcal{L}(j) &= \frac{1}{\mathcal{W}} \sum_{i=j-(n-k)}^{\mathcal{M}} \frac{i}{k} \\
&= \left(\frac{1}{\mathcal{M} - j + (n-k) + 1} \right) \left(\frac{1}{k} \right) \\
&\quad \left(\frac{\mathcal{M}(\mathcal{M}+1)}{2} - \frac{(j-(n-k))(j-(n-k)-1)}{2} \right). \quad (4.24)
\end{aligned}$$

It is now easy to derive $\mathcal{L}_{sp}$, the average data packet loss rate (with the use of

FEC) for the SP approach as follows:

$$\begin{aligned}
\mathcal{L}_{sp} &= \sum_{j=n-k+1}^{n} P^{(1)}(j,n)\mathcal{L}(j) \\
&= \sum_{j=n-k+1}^{n} \left[P_0^{(1)}(j,n)\pi_0(1) + P_1^{(1)}(j,n)\pi_1(1)\right]\mathcal{L}(j) \\
&= \sum_{j=n-k+1}^{n} \left[\left(L_0(j,n) + H_0(j,n)\right)\left(\frac{\mu_1(1)}{\mu_0(1)+\mu_1(1)}\right)\mathcal{L}(j) + \right. \\
&\qquad \left. \left(L_1^{(1)}(j,n) + H_1^{(1)}(j,n)\right)\left(\frac{\mu_0(1)}{\mu_0(1)+\mu_1(1)}\right)\mathcal{L}(j)\right]. \quad (4.25)
\end{aligned}$$

To derive the average data packet loss rate (with use of FEC) for the MP approach, let us first consider a simple case of dual-path streaming. Assume that there are two servers S_1 and S_2 that use two different, possibly heterogeneous, paths. We use the same FEC scheme as described above to generate a stream of data divided into n packet FEC groups. To transmit the packets within a FEC group, server S_1 transmits n_1 packets while server S_2 transmits n_2 packets such that $n_1 + n_2 = n$. Based on the similar argument we made above in the SP case, we have

$$P^{(1)}(j,n_1) = P_0^{(1)}(j,n_1)\pi_0(1) + P_1^{(1)}(j,n_1)\pi_1(1) \quad j = 0,1,\ldots,n_1 \quad (4.26)$$

$$P^{(2)}(j,n_2) = P_0^{(2)}(j,n_2)\pi_0(2) + P_1^{(2)}(j,n_2)\pi_1(2) \quad j = 0,1,\ldots,n_2 \quad (4.27)$$

The computation of $P_i^{(h)}(j,n_h)$ where $i \in \{0,1\}$ and $h \in \{1,2\}$ is similar to the approach mentioned above, that is, by evaluating the entries of the corresponding four matrices. The computational complexity would then be $\Theta(4(n_1+1)^2 + 4(n_2+1)^2)$.

Let P_{2p} be the probability of an irrecoverable error within a FEC group. It is equal to

$$P_{2p} = \sum_{j=n-k+1}^{n} \sum_{h=0}^{j} P^{(1)}(h,n_1)P^{(2)}(j-h,n_2), \quad (4.28)$$

which involves a convolution between the two probability mass functions, $P^{(1)}(j, n_1)$ and $P^{(2)}(j, n_2)$. Let $\mathcal{L}_{2p}$ be the average data packet loss rate (with use of FEC) for the dual path approach. Then, we have

$$\mathcal{L}_{2p} = \sum_{j=n-k+1}^{n} \sum_{h=0}^{j} P^{(1)}(h, n_1) P^{(2)}(j-h, n_2) \mathcal{L}(j). \quad (4.29)$$

In general, if we employ N servers $S_1, S_2, \ldots, S_N$, then the probability of an irrecoverable error within a FEC group is

$$P_{Np} = \sum_{j=n-k+1}^{n} \left(\sum_{i_1+\ldots+i_N=j} P^{(1)}(i_1, n_1) P^{(2)}(i_2, n_2) \cdots P^{(N)}(i_N, n_1) \right). \quad (4.30)$$

The average data packet loss rate with FEC under a MP streaming with N paths is

$$\mathcal{L}_{Np} = \sum_{j=n-k+1}^{n} \left(\sum_{i_1+\ldots+i_N=j} P^{(1)}(i_1, n_1) P^{(2)}(i_2, n_2) \cdots P^{(N)}(i_N, n_1) \right) \mathcal{L}(j). \quad (4.31)$$

In the case of the other two performance measures, namely, the conditional burst length distribution and the lag-1 autocorrelation, we resort to the use of simulation, as described in the following section.

4.2 Analytical Model Based Evaluation

In this section, we further evaluate the loss characteristics of the SP vs. MP methods using simulations of the Gilbert model described in Section 4.1.1. The simulations allow us to consider the loss characteristics under more sophisticated scenarios than in Section 4.1.1. Specifically, we assume a MPEG-1 video streaming application which generates packets at a rate of 120 packets per second with each packet containing 1400 bytes. We consider at most three senders (S_1, S_2, S_3) and one receiver C. Sender S_i uses path i to transmit its fraction of the data; unless otherwise stated, these paths are assumed to be independent. Moreover, in the figures given below (unless otherwise stated), the

curves corresponding to SP streaming use path 1, the curves corresponding to MP streaming with 2 senders use paths 1 and 2, and the curves corresponding to MP streaming with 3 senders use all three paths. Unless stated otherwise, the packet assignment is carried out in a round-robin manner, e.g., if we use all three senders, then sender S_i transmits data packets at a rate of 40 packets per second. The loss process of path i is modeled by a continuous stationary Gilbert model (as defined in Section 4.1.1). Unless stated otherwise, we use $\mu_0(i) = 20$ and $\mu_1(i) = 70$, for $i = 1, 2, 3$. Lastly, we consider all the same performance metrics as defined in Section 4.1.1.

Experiment 1 (Data Loss Rate): In this experiment, we study the data packet loss rate of the SP and MP approaches, using only two paths, 1 and 2. The path parameters are as described above except that we vary the $\mu_0(2)$ parameter from 5 to 50. Table 4.1 illustrates the data loss rate for the single path(s) and the dual-path approaches (in each case, with and without the use of FEC, where the parameters for the FEC scheme are $n = 5$ and $k = 4$). We can observe that in this experiment:

- Without the use of FEC, the data packet loss rate of the dual path is approximately the mean of the data packet loss rates of paths 1 and 2. These results are consistent with the derivation of Section 4.1.1.
- With the use of FEC, (in this case $n = 5$ and $k = 4$), the achieved data packet loss rate can be *less* than the average of the data packet loss rates of the two corresponding single paths. This may occur due to the fact that error burst lengths in dual-path streaming tend to be shorter than in single-path streaming (refer Theorem 4.4 in Section 4.1.1), and hence a chance of recovery of lost data (using FEC) should also be higher.

This experiment also illustrates the potential advantages of multi-path streaming over "best path" streaming, even when losses (rather than throughput) are the important consideration. That is, when multiple paths are available

(but throughput is not the issue), another approach might be to stream the data over the "best" available path (and as congestion conditions change keep switching the streaming of the data to the best available path at the time). Our experiment shows that MP streaming could provide better loss characteristics (e.g., when FEC is used) than the "best" available path. (Please refer to Experiment 6 below on further comparison to a best-path type approach.)

Loss rate: ($\mu_0(2)$)	single path: path 1 w/o FEC	single path: path 2 w/o FEC	dual-path without FEC	single path: path 1 with FEC	single path: path 2 with FEC	dual-path with FEC
5	0.221743	0.066767	0.144351	0.189053	0.053048	0.101264
15	0.221743	0.176153	0.199395	0.189053	0.147171	0.141632
20	0.221743	0.221743	0.222255	0.189053	0.189053	0.158861
35	0.221743	0.332848	0.278178	0.189053	0.297647	0.201947
50	0.221743	0.416609	0.319230	0.189053	0.385602	0.235681

Table 4.1: Data loss rate with heterogeneous paths.

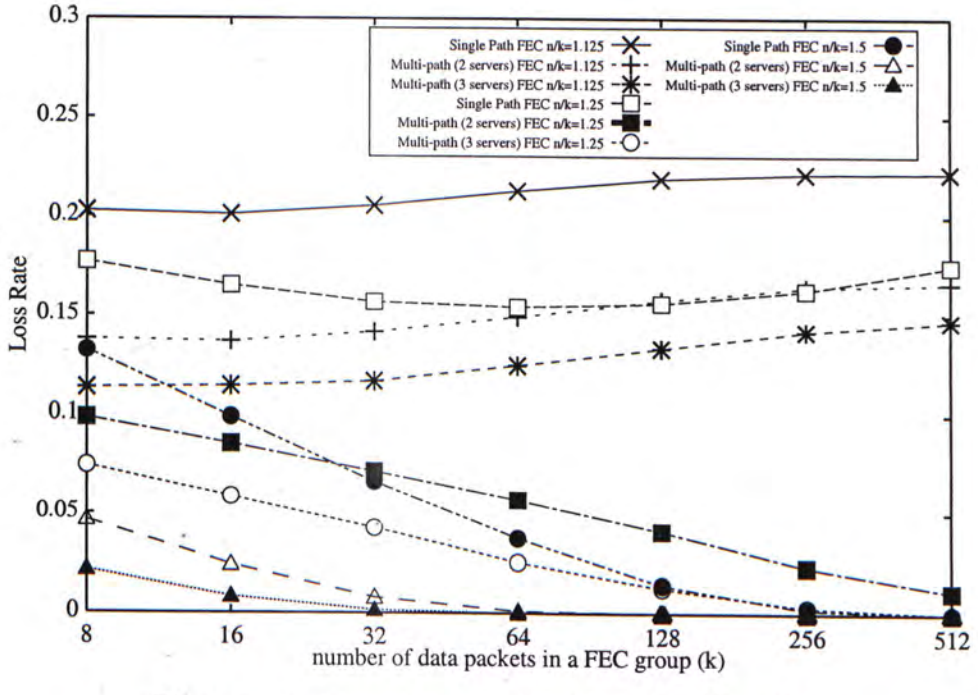


Figure 4.2: Loss rate as a function of n/k and k

Experiment 2 (Data Loss Rate as a function of FEC parameters): In this experiment, we study the effects of FEC parameters on the data loss rate.

In general, there are two ways to vary the FEC parameters. We can:

1. Increase the degree of redundancy (e.g., for a given value of k, increase the value of n). Note that by increasing the degree of redundancy, we also increase the amount of traffic on the network.

2. Increase the values of n and k but keep the same ratio of n/k. This implies that we increase the FEC group size, and hence the application needs to maintain a larger receiving buffer (for reconstruction purposes in case of loss) as well as experience potentially higher latency (since a larger amount of information must be received prior to reconstruction of missing information).

Figure 4.2 illustrates the effects of FEC parameters on the data loss rate, and specifically, it depicts data loss rates for SP and MP streaming with $n/k = 1.125, 1.25$ and 1.5 as well as with different FEC group sizes (where we vary the number of data packets in a FEC group (k) from 8 to 512 packets). In this case the path parameters are $\mu_0(1) = 20$, $\mu_1(1) = 70$, $\mu_0(2) = \mu_0(3) = 10$, and $\mu_1(2) = \mu_1(3) = 80$. We observe that:

- Increasing the amount of redundancy (e.g., from $n/k = 1.125$ to 1.5) in SP or MP streaming can reduce the data loss rate. However, one can achieve a lower data packet loss rate with MP streaming with a smaller n/k ratio (as compared to SP streaming). In other words, without introducing additional network traffic, we can obtain better performance with MP streaming.

- Increasing the number of data packets in a FEC group (while keeping the same ratio of n/k) may not necessary reduce the data loss rate. For example, consider SP streaming; as we increase k, the data loss rate actually increases in some cases. The maybe explained by a possible

"convergence" of the data loss rate, as a function of n and k, to a non-zero value (please refer to the Appendix for details).

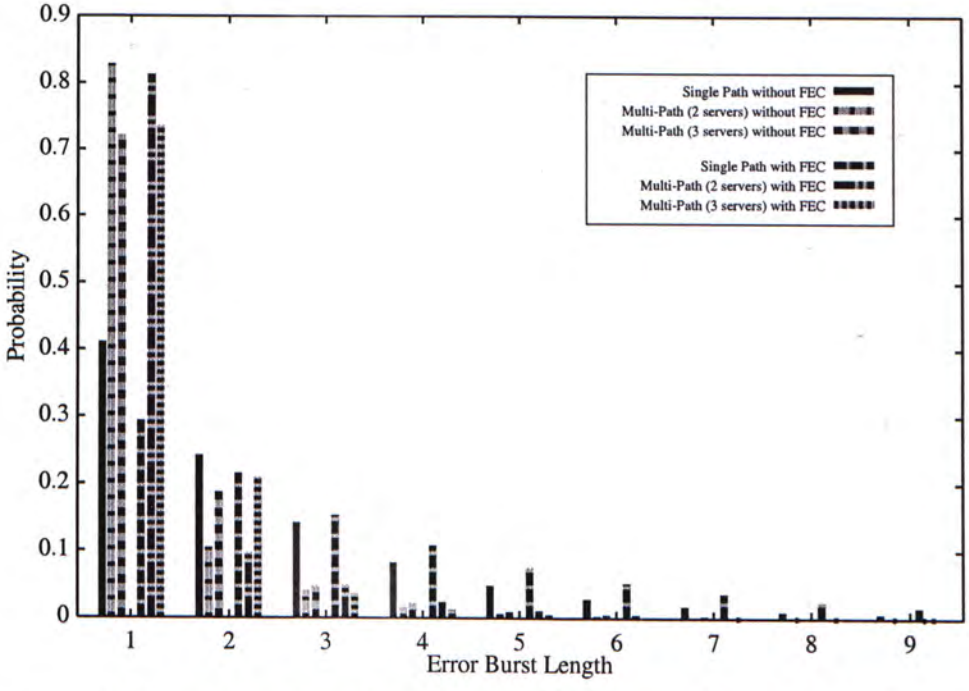


Figure 4.3: Conditional probability mass functions of error burst length.

Experiment 3 (Conditional Error Burst Length): In this experiment, we compare the conditional burst length distribution, conditioned on there being at least one error. Figure 4.3 illustrates the conditional probability mass functions of error burst length (as defined in Section 4.1.1). In this experiment, we observe that the packet error burst length is indeed stochastically less than the error burst length of the single path streaming. We also note, that the condition of Theorem 4.4 in Section 4.1.1 holds in this experiment[3] . This relationship also holds when we employ FEC.

Experiment 4 (Lag-1 Autocorrelation): In this experiment, we study the lag-1 autocorrelation of packet losses for both SP and MP streaming (as defined in Section 4.1.1). Figure 4.4 illustrates the lag-1 autocorrelation where

[3]Note that here we illustrate the probability mass function rather than the probability distribution function, as we believe it depicts the results of the experiment better.

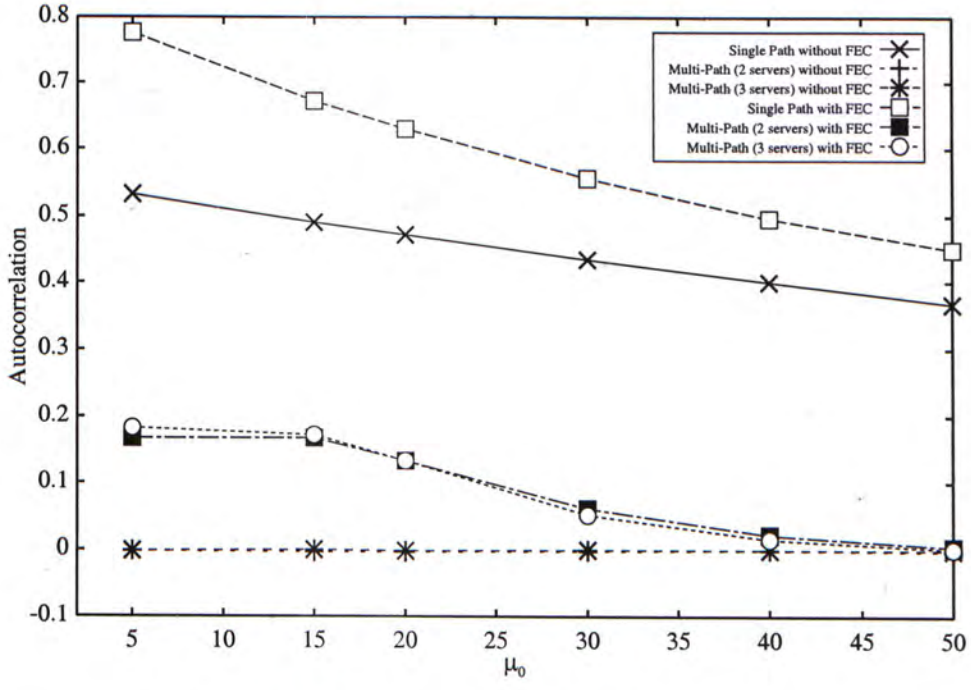


Figure 4.4: Lag-1 autocorrelation.

$\mu_1(1) = \mu_1(2) = \mu_1(3) = 70$ and $\mu_0(i)$ is varied (identically) for all three paths. We make the following observations.

- When we use MP streaming without FEC, the lag-1 autocorrelation is nearly zero while the lag-1 autocorrelation of SP path streaming (with or without FEC) can be highly correlated.

- The use of FEC may increase the lag-1 autocorrelation (for both SP and MP approaches). This may be explained as follows. The irrecoverable losses (after the error correction process) are likely to end up "closer" in the resulting data stream than in the original data stream (one without the use of erasure codes), and hence the lag-1 autocorrelation in this new stream behaves similarly to lag-h autocorrelation of the original stream, where $h > 1$. However, we still observe that the lag-1 autocorrelation of MP streaming is significantly closer to zero as compared to SP streaming, even with the use of FEC.

Experiment 5 (Effects of Load Distribution Among Senders): In previous experiments, all senders transmitted packets in a round-robin manner and hence the load distribution between all the senders was the same. In this experiment, we investigate effects of load distribution among senders. Specifically, we distribute the load among two senders only, where parameter α refers to the fraction of packets sent by sender 1. For instance, when $\alpha = 0.3$, sender 1 sends 30% of the packets while sender 2 sends 70% of packets. In the cases of $\alpha = 0$ and $\alpha = 1$, this degenerates to single path streaming using path 1 and path 2, respectively. Both path 1 and path 2 have the same parameters with $\mu_0 = 5, 20$, or 40 and μ_1 fixed at 70. Figure 4.5 illustrates results of this experiment. We observe that there is a slight improvement in loss rate when FEC is used and the load is equally distributed between the two senders. Moreover, in this experiment, the lag-1 autocorrelation reaches its minimum value under equal load distribution. This implies that simple round-robin packet distribution among paths should result in a higher quality of received video. That is, this simple approach of equal distribution is fairly robust.

Experiment 6 (Sensitivity Analysis): In this experiment, we study the relative performance of MP streaming vs. SP streaming when the SP streaming is performed over the *best* of the available paths. For example, if the performance metric is loss rate, then the path with the lowest loss rate is used. We note that implementation of this form of *best* single path streaming would likely require a fairly accurate monitoring of the loss characteristics of a path; otherwise, the wrong path might be selected. That is, the sensitivity (or robustness) of the streaming decisions to the accuracy of the available information about the network is an important issue.

In this sensitivity experiment, we consider a two-path system, where the fixed parameters are $\mu_0(1) = 20$ and $\mu_1(1) = \mu_1(2) = 70$ and $\mu_0(2)$ is varied from 5 to 50. In this scenario, the best-path approach believes (based on

collected measurements) that path 2 is the better path (e.g., it may mis-estimate the $\mu_0(2)$ parameter as being less than 20). We vary $\mu_0(2)$ from 5 to 50, in order to see the effect of mis-estimation; hence, the best path approach *over-estimates* this parameter when the real value of $\mu_0(2)$ is less than 20 and *under-estimates* this parameter when the real value of $\mu_0(2)$ is greater than 20. We also consider a very *simple* MP streaming approach, where the load is distributed equally among the two senders in a round-robin manner (i.e., odd-numbered packets are sent along path 1 while even-numbered packets are sent along path 2).

Figure 4.6 illustrates the *relative* loss rate (using several different FEC schemes) of the two approaches, which is defined as the data loss rate of dual-path streaming divided by the data loss rate of best-path streaming. Hence, a relative loss of less than 1 implies that the simple dual-path approach is doing better than the best path approach. In this figure, we observe that simple dual-path (round-robin) streaming does quite well compared to best-path streaming, even when there is significant differences in loss characteristics between the two paths. Of course, in cases where the best path has much better loss characteristics and with relatively little redundant information, the best-path approach has a lower data loss rate. However, we note that the best-path approach would require relatively accurate estimation of the path characteristics, which may be non-trivial especially as network conditions change. Hence, we believe that the MP approach is more robust as compared to best-path streaming.

Experiment 7 (Effects of Shared Points of Congestion on Various Performance Metrics): In this experiment, we study the effects of *shared points-of-congestion*, between the paths used by the different senders, on various performance measures. Senders S_1 and S_2 share the same point-of-congestion, which we can characterize by a Gilbert model (as defined in Section 4.1.1). Sender S_3 uses a path which does not share a point of congestion

with S_1 and S_2 (as before, this path is characterized by a Gilbert model). All the application settings remain the same, and we consider the following four configurations.

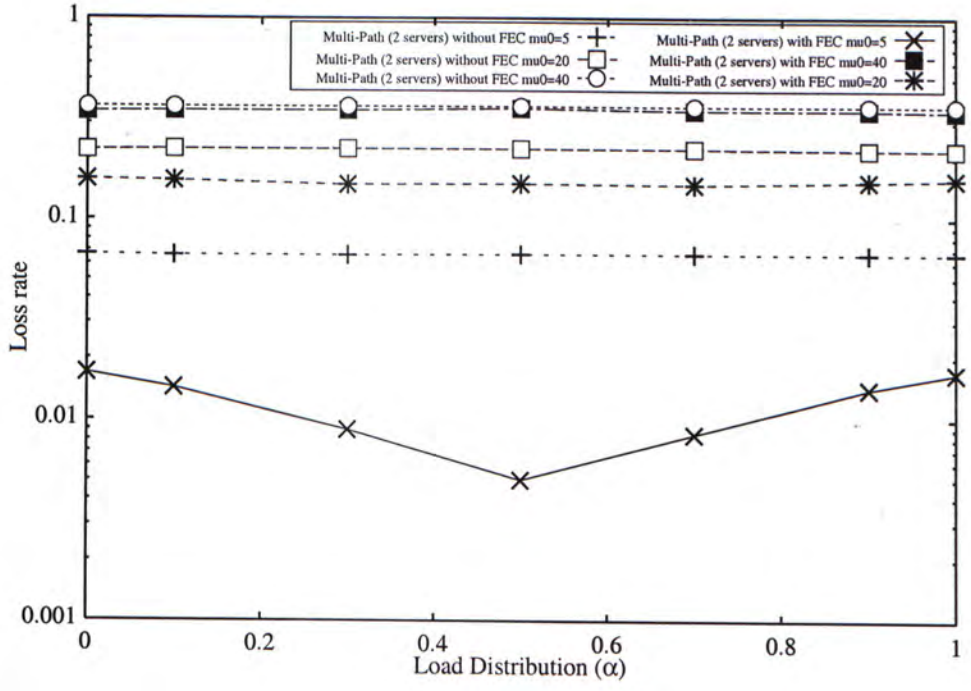
- **Configuration 1:** Sender 1 is the only one streaming the data.
- **Configuration 2:** Senders 1 and 3 stream the data in a round-robin manner, i.e., each transmits at a rate of 60 data packets/second.
- **Configuration 3:** Senders 1, 2, and 3 stream the data in a round-robin manner, i.e., each transmits at a rate of 40 data packets/second.
- **Configuration 4:** Senders 1, 2, and 3 stream the data, but senders 1 and 2 transmit at a rate of 20 data packets/second while sender 3 transmits at a rate of 80 data packets/second.

Figure 4.7 illustrates the data loss rate and lag1-autocorrelation for above configurations, when FEC is used, with $(n = 10, k = 8)$. Moreover, we vary the $\mu_0(1), \mu_1(1), \mu_0(3)$, and $\mu_1(3)$ parameters (as described in the figure). From this figure, we observe the following.

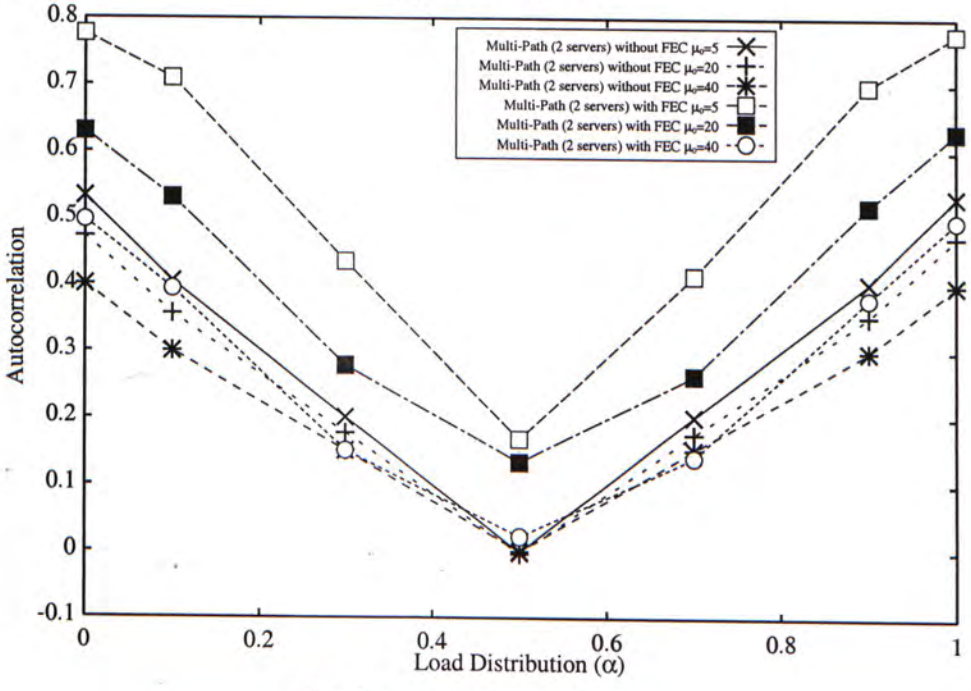
- MP streaming (configuration 2, 3, and 4) has a lower data loss rate as compared to SP streaming (configuration 1).
- Detecting shared points of congestion is important, as including a greater number of paths in a transmission (under such conditions) may adversely affect the data loss rate. For example, equally splitting the workload among senders 1 and 3 (configuration 2) achieves a lower data loss rate than equally splitting the workload among senders 1, 2, and 3 (configuration 3). This occurs because senders 1 and 2 share the same point of congestion and with configuration 3 we are actually sending a greater fraction of the workload through this shared point of congestion. This agrees with intuition, as in this section we are effectively modeling a

shared point of congestion as a single path/bottleneck, i.e., configuration 3 effectively corresponds to a configuration with two senders and an unequal split of workload between them.

- Of course, shared points of congestion adversely affect the lag-1 autocorrelation metric. For example, configuration 3 has a higher lag-1 autocorrelation than configuration 2. Again, the explanation given in the preceding point applies here as well.



(a) Loss Rate



(b) Lag-1 Autocorrelation

Figure 4.5: Loss rate and lag-1 autocorrelation for different load distributions for the dual-path streaming

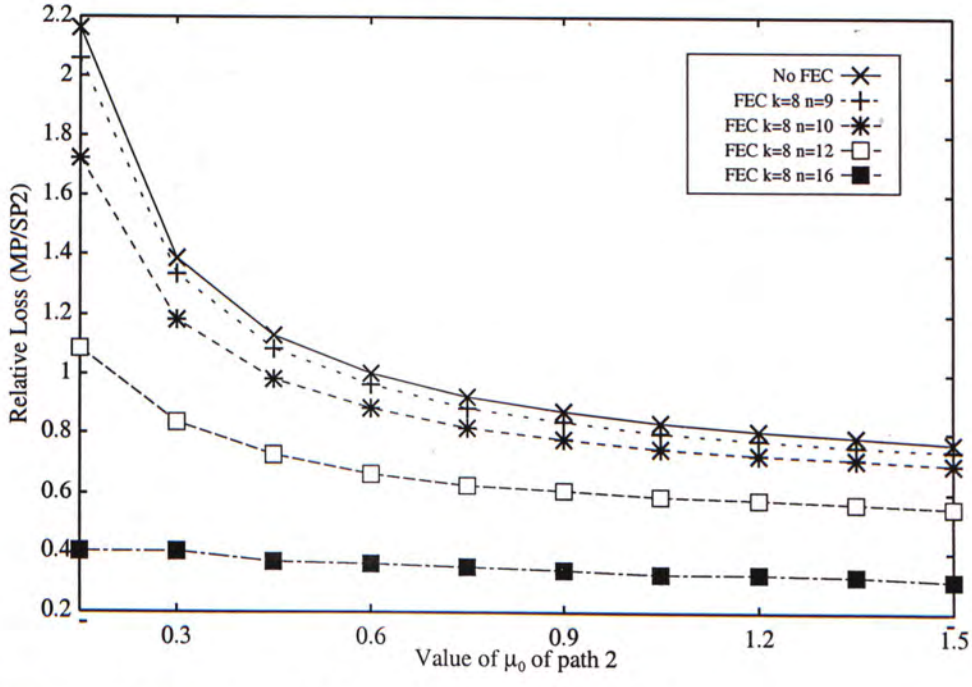
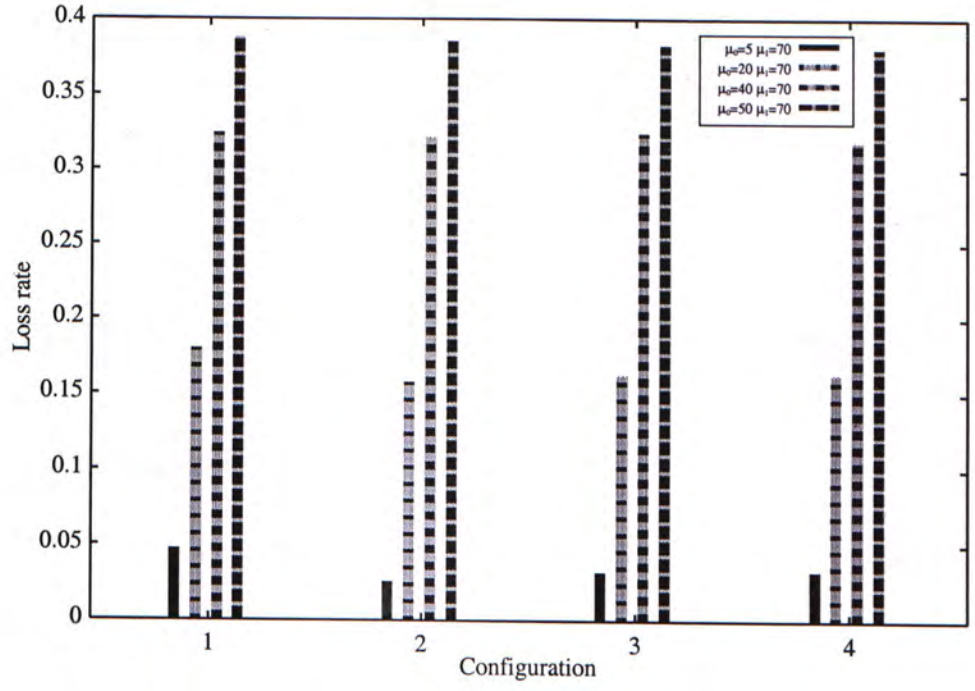
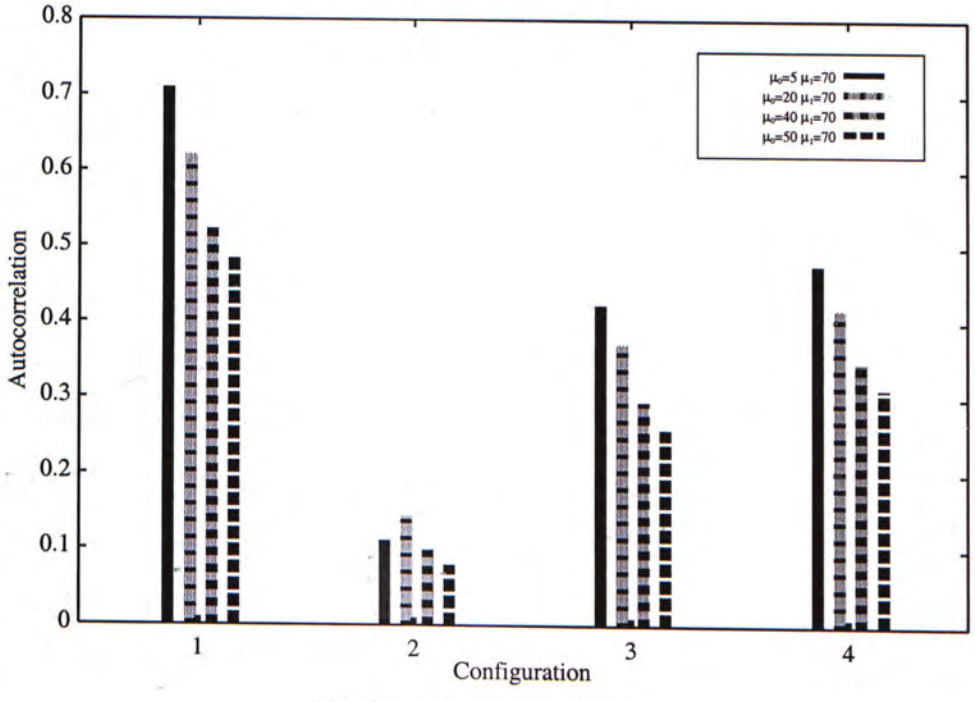


Figure 4.6: Relative loss of dual-path vs. single path when we vary $\mu_0(2)$ and FEC group size.



(a) Data Loss Rate



(b) Lag1-autocorrelation

Figure 4.7: Effects of shared points-of-congestion on data loss rate and lag-1 autocorrelation with FEC ($n = 10, k = 8$).

Chapter 5

Functional Gilbert Model and Optimization

5.1 Functional Gilbert Model

In Chapter 4, we illustrated that one can obtain improvements in (a) loss rate, (b) lag-1 autocorrelation, and (c) lost packets burst length by employing multi-path streaming techniques. However, the above results were shown under the conventional Gilbert model One major limitation of using a conventional Gilbert model is that the loss process of a path is *independent* of the bandwidth requirements of the streaming continuous media application. In other words, the loss rate as viewed by the receiver is fixed, independent of the sending rate of the application.

To test whether the loss rate depends on the application's bandwidth requirements, we carry out the following Internet experiment. We transmit UDP packets from Asia to the USA, where each packet has a size of 1400 bytes. The application sends packets at various rates, from 120 pkt/sec (around 1.34 Mbps) to 1200 pkt/sec (around 12.8 Mbps), with a step size of 120 pkt/sec interval. For each sending rate, the streaming experiment is carried out for 6 minutes, and we measure the corresponding achieved loss rate at the receiver. Table 5.1 illustrates the achieved loss rate for each experimental setting, which

is the *fraction* of lost packets as measured at the receiver.

Application's Sending Rate (pkt/sec)	**Measured Loss Rate**
120	0.052361
240	0.050729
360	0.069468
480	0.083385
600	0.098222
720	0.106852
840	0.121845
960	0.192457
1080	0.300455
1200	0.375840

Table 5.1: Data loss rate vs. different sending rates.

Table 5.1 supports our hypothesis that the conventional Gilbert model may not be sufficient for characterizing the loss process of a path, i.e., when there is a strong correlation between the application's bandwidth requirements and the loss characteristics. Since the above evidence suggests that the application's sending rate can significantly affect the achieved loss rate, we propose to use a *functional Gilbert model* as a general approach to characterizing the bursty loss nature of a path as well as its dependency on the application's bandwidth requirements.

Let λ denote the application's average sending rate, in units of packets per second. For a stationary continuous time functional Gilbert model, the packet loss process along path k is described by a two state continuous time Markov chain $\{X_k(t)\}$ where $X_k(t) \in \{0, 1\}$. Figure 5.1 depicts the state transition diagram of this functional Gilbert model. Similarly to the previous definition, if a packet is transmitted at time t when the state of path k is $X_k(t) = 0$, then no packet loss occurs. On the other hand, the transmitted packet is considered lost if $X_k(t) = 1$. The transition rate from state 0 to state 1 takes

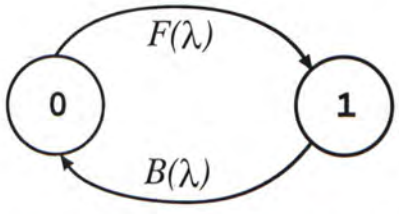


Figure 5.1: Functional Gilbert model: transition rates are functions of λ.

a *"functional"* form of $\mathcal{F}(\lambda)$, which we assume to be a continuous and *non-decreasing* function of λ. The transition rate from state 1 to state 0 takes a *"functional"* form of $\mathcal{B}(\lambda)$, which we assume to be a continuous and *non-increasing* function of λ. We note that intuitively it makes sense for $\mathcal{F}(\lambda)$ to be a non-decreasing function of λ; similarly, it makes intuitive sense for $\mathcal{B}(\lambda)$ to be a *non-increasing* function of λ. Hence, in practice, these assumptions should not be restrictive.

When one uses a functional Gilbert model to characterize the loss process of a path, we have the following result.

Theorem 5.1 Let there be $M \geq 1$ homogeneous paths available for continuous media streaming. Define $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_M]$ as the vector which determines how the traffic is split among these M paths, where $\alpha_i \geq 0$ and $\sum_i^M \alpha_i = 1$. Then the achieved loss rate via the M-path streaming approach (P_M) is less than or equal to the achieved loss rate via the single path streaming approach (P_1), for *all* possible valid traffic splitting vectors $\boldsymbol{\alpha}$.

Proof: Let P_M be the achieved loss rate for the M-path streaming approach, with all paths being homogeneous. We can express P_M as

$$P_M = \sum_{j=1}^{M} \alpha_j \frac{\mathcal{F}(\alpha_j \lambda)}{\mathcal{F}(\alpha_j \lambda) + \mathcal{B}(\alpha_j \lambda)}.$$

Since the paths are homogeneous, the achieved loss rate under the single path streaming approach is

$$P_1 = \frac{\mathcal{F}(\lambda)}{\mathcal{F}(\lambda) + \mathcal{B}(\lambda)}.$$

Let path k^* be the path that has the largest loss rate, that is,

$$\frac{\mathcal{F}(\alpha_{k^*}\lambda)}{\mathcal{F}(\alpha_{k^*}\lambda)+\mathcal{B}(\alpha_{k^*}\lambda)} \geq \frac{\mathcal{F}(\alpha_j\lambda)}{\mathcal{F}(\alpha_j\lambda)+\mathcal{B}(\alpha_j\lambda)} \qquad \text{for all } j \in \{1,\ldots,M\}.$$

Then we have:

$$P_M \leq \sum_{j=1}^{M} \alpha_j \frac{\mathcal{F}(\alpha_{k^*}\lambda)}{\mathcal{F}(\alpha_{k^*}\lambda)+\mathcal{B}(\alpha_{k^*}\lambda)} = \frac{\mathcal{F}(\alpha_{k^*}\lambda)}{\mathcal{F}(\alpha_{k^*}\lambda)+\mathcal{B}(\alpha_{k^*}\lambda)} = P_M^*.$$

Note that $P_M^* - P_1 \leq 0$ implies that $P_M - P_1 \leq 0$. Since $P_M^* - P_1 = \frac{\mathcal{F}(\alpha_{k^*}\lambda)}{\mathcal{F}(\alpha_{k^*}\lambda)+\mathcal{B}(\alpha_{k^*}\lambda)} - \frac{\mathcal{F}(\lambda)}{\mathcal{F}(\lambda)+\mathcal{B}(\lambda)}$, if $\mathcal{F}(\alpha_{k^*}\lambda)\,(\mathcal{F}(\lambda)+\mathcal{B}(\lambda)) - \mathcal{F}(\lambda)\,(\mathcal{F}(\alpha_{k^*}\lambda)+\mathcal{B}(\alpha_{k^*}\lambda)) \leq 0$, then $P_M^* - P_1 \leq 0$. Expanding the terms, we have:

$$\begin{aligned}
& \mathcal{F}(\alpha_{k^*}\lambda)\mathcal{F}(\lambda) + \mathcal{F}(\alpha_{k^*}\lambda)\mathcal{B}(\lambda) - \mathcal{F}(\lambda)\mathcal{F}(\alpha_{k^*}\lambda) - \mathcal{F}(\lambda)\mathcal{B}(\alpha_{k^*}\lambda) \\
= \;& \mathcal{F}(\alpha_{k^*}\lambda)\mathcal{B}(\lambda) - \mathcal{F}(\lambda)\mathcal{B}(\alpha_{k^*}\lambda) \\
\leq \;& \mathcal{F}(\alpha_{k^*}\lambda)\mathcal{B}(\alpha_{k^*}\lambda) - \mathcal{F}(\lambda)\mathcal{B}(\alpha_{k^*}\lambda) \\
= \;& \mathcal{B}(\alpha_{k^*}\lambda)\,[\mathcal{F}(\alpha_{k^*}\lambda) - \mathcal{F}(\lambda)] \leq 0.
\end{aligned}$$

Remark: The implication of the above theorem is that under the homogeneous path assumption, *any* valid traffic splitting in a multi-path streaming approach will do no worse than single path streaming in terms of the packet loss rate metric. Of course, an interesting question to ask is what is the *right metric* to optimize in determining the traffic split among these M servers. We consider this question and the resulting effects on the system's performance below.

5.2 Optimal Traffic Splitting

In this section, we present a framework for determining the right traffic splitting between the multiple paths used for streaming of continuous media, based on an optimization of one of the performance metrics defined above (please refer to Section 5.1). Although, we consider a single performance metric (at a time) in this optimization process, in Section 5.3 we illustrate the effects of this optimization process on the other performance metrics.

5.2.1 Optimization Based on Achieved Loss Rate

We first consider the minimization of the loss rate, P_M, achieved at the receiver. For path j, let $\mathcal{F}_j(b)$ denote the functional transition rate from state 0 to state 1 when the streaming traffic on path j is b pkt/sec. Similarly, $\mathcal{B}_j(b)$ denotes the functional transition rate from state 1 to state 0 when the streaming traffic on path j is b pkt/sec.

Let us first consider a simple case wherein there are two paths available for continuous media streaming. We define $F_j(\alpha_j\lambda) = \frac{\mathcal{F}_j(\alpha_j\lambda)}{\mathcal{F}_j(\alpha_j\lambda)+\mathcal{B}_j(\alpha_j\lambda)}$, for $j = 1, 2$. We can express the achieved loss rate as

$$P_2 = \alpha_1 F_1(\alpha_1\lambda) + (1-\alpha_1)F_2((1-\alpha_1)\lambda).$$

To find the optimal traffic split $\boldsymbol{\alpha}^* = [\alpha_1^*, 1-\alpha_1^*]$, we set the first derivative of P_2 to zero:

$$\begin{aligned}\frac{dP_2}{d\alpha_1} &= F_1(\alpha_1\lambda) + \alpha_1\lambda F_1'(\alpha_1\lambda) - \\ &\quad F_2((1-\alpha_1)\lambda) - (1-\alpha_1)\lambda F_2'((1-\alpha_1)\lambda) = 0. \qquad (5.1)\end{aligned}$$

Solution of the above equation can be carried out by using standard numerical methods for finding roots.

It is interesting to note that, when the two paths are homogeneous (i.e., $F_1()$ and $F_2()$ have the same form), then equal splitting of traffic along these two paths (i.e., $\alpha_1 = \alpha_2 = 1/2$) is a critical point in the above optimization problem. This claim can be easily verified as follows:

$$F_1(\lambda/2) + \frac{\lambda}{2}F_1'(\lambda/2) - F_2(\lambda/2) - \frac{\lambda}{2}F_2'(\lambda/2) = 0.$$

In general, when we have $M \geq 1$ paths, we can formulate the following constrained optimization problem:

$$\min_{\boldsymbol{\alpha}} P_M = \sum_{j=1}^{M} \alpha_j \frac{\mathcal{F}_j(\alpha_j\lambda)}{\mathcal{F}_j(\alpha_j\lambda)+\mathcal{B}_j(\alpha_j\lambda)} \qquad \text{s.t.} \sum_{j=1}^{M}\alpha_j = 1. \qquad (5.2)$$

One can reformulate this constrained optimization problem using the Lagrangian multiplier method as

$$L(P_M, \phi) = \sum_{j=1}^{M} \alpha_j \frac{\mathcal{F}_j(\alpha_j \lambda)}{\mathcal{F}_j(\alpha_j \lambda) + \mathcal{B}_j(\alpha_j \lambda)} + \phi \left(1 - \sum_{j=1}^{M} \alpha_j \right). \tag{5.3}$$

The solution to the minimization of the Lagrangian function in Equation (5.3) is equivalent to the solution to the constrained optimization in Equation (5.2). The necessary conditions for obtaining the minimum value of $L(P_M, \phi)$ are

$$\frac{\partial L(P_M, \phi)}{\partial \alpha_j} = 0 \qquad \text{for } j = 1, \ldots, M, \tag{5.4}$$

$$\frac{\partial L(P_M, \phi)}{\partial \phi} = 0. \tag{5.5}$$

This is equivalent to solving a system of $M+1$ equations, possibly non-linear, wherein multiple solutions may exist. In general, one has to rely on numerical solution techniques to find the critical points and the optimal solution.

When all M paths are homogeneous, we have the following result.

Theorem 5.2 If all M-paths are homogeneous, then the traffic splitting vector $\boldsymbol{\alpha} = [1/M, \cdots, 1/M]$ satisfies Equations (5.4) and (5.5).

Proof: Define $F_j(\alpha_j \lambda) = \frac{\mathcal{F}_j(\alpha_j \lambda)}{\mathcal{F}_j(\alpha_j \lambda) + \mathcal{B}_j(\alpha_j \lambda)}$. Consider the partial derivative with respect to α_j, for $j = 1, 2, \ldots, M$; we then have

$$\frac{\partial L(P_M, \phi)}{\partial \alpha_j} = F_j(\alpha_j \lambda) + \alpha_j F_j'(\alpha_j \lambda)\lambda - \phi = 0.$$

We can relate these M equations as

$$F_1(\alpha_1 \lambda) + \alpha_1 F_1'(\alpha_1 \lambda)\lambda = \cdots = F_M(\alpha_M \lambda) + \alpha_M F_M'(\alpha_M \lambda)\lambda.$$

Since all M paths are homogeneous (i.e., all $F_j()$ functions are the same), then one obvious solution to the equation above is $\boldsymbol{\alpha} = [1/M, 1/M, \cdots, 1/M]$. In other words, equal splitting of traffic among M servers provides a critical point for the constrained optimization problem in Equation (5.2).

To illustrate the performance gains due to multi-path streaming, we consider a family of functional transition rates. In particular, the functional Gilbert model for path j, for $j = 1, \ldots, M$, has the form of

$$\mathcal{F}_j(b) = \beta_j (b/\sigma_j)^{\theta_j} + S_j, \tag{5.6}$$

$$\mathcal{B}_j(b) = \kappa_j e^{-\chi_j b} + \varphi_j \tag{5.7}$$

where $\beta_j, \kappa_j, \chi_j \geq 0$ and $\sigma_j, \theta_j, S_j, \varphi_j > 0$. In other words, $\mathcal{F}_j$ is a non-decreasing function while $\mathcal{B}_j$ is an exponential decreasing function of the traffic bandwidth b.

It is important to point out that the form of functional transition rates listed above can represent a large family of Gilbert models. For example, for a conventional Gilbert model, a constant transition rate from state 0 to state 1 can be represented by setting $S_j > 0$, $\beta_j = 0$, $\sigma_j > 0$, and $\theta_j = 0$. A constant transition rate from state 1 to state 0 can be represented by setting $\kappa_j = \chi_j = 0$ and $\varphi_j > 0$.

Consider now an application with a bandwidth requirement of 360 pkt/sec (or around 3.84 Mbps). We first consider a system with homogeneous paths wherein the parameters for $\mathcal{F}(b)$ are $\beta = 0.3$, $\sigma = 10$, $\theta = 0.5$, $S = 0.01$ and the parameters for $\mathcal{B}(b)$ are $\kappa = 10$, $\chi = 1/1500$, $\varphi = 0.1$. Figure 5.2 illustrates the loss rates for a system under two homogeneous paths as a function of α_1 (the corresponding α_2 is $1 - \alpha_1$). Figure 5.3, on the other hand, illustrates the contour map of loss rates for three homogeneous paths (with the same parameters as the two paths system) at various values of $\boldsymbol{\alpha}$. It is interesting to observe that under this homogeneous path example, the loss rate using single path streaming is around 18.5%. Under the 2-paths streaming approach, the achieved loss rate is reduced to 12.5%, with $\boldsymbol{\alpha}^* = [0.5, 0.5]$. A 3-paths system can reduce the loss rate further to 10.1%, with $\boldsymbol{\alpha}^* = [1/3, 1/3, 1/3]$. It is also interesting to point out that for this homogeneous paths system, the loss rate is a convex function and we can easily find a global optimal traffic splitting

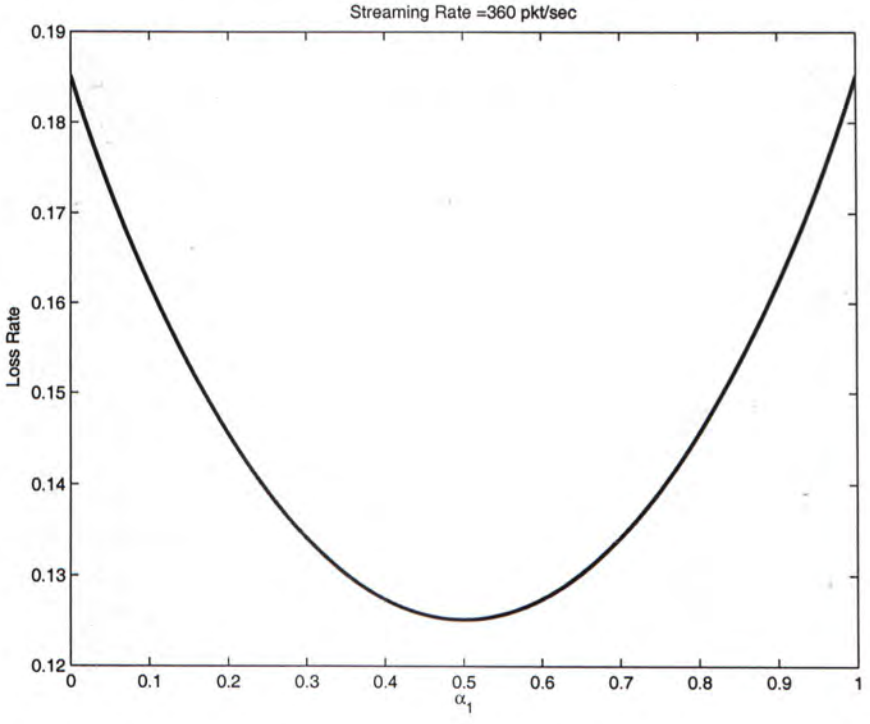


Figure 5.2: Loss rates under 2 homogeneous paths, with $\boldsymbol{\alpha}^* = [0.5, 0.5]$ and corresponding loss rate of 12.5%.

$\boldsymbol{\alpha}^*$.

Now let us consider an application with the same bandwidth requirement of 360 pkt/sec (or around 3.84 Mbps) but under the heterogeneous paths setting. We first consider a system with two heterogeneous paths wherein $\mathcal{F}_1(b) = 0.25\,(b/10)^{0.5} + 0.01$, $\mathcal{F}_2(b) = 0.35\,(b/10)^{0.5} + 0.01$, $\mathcal{B}_1(b) = 15e^{-b/1500} + 0.1$, and $\mathcal{B}_2(b) = 5e^{-b/1500} + 0.1$. Figure 5.4 illustrates the loss rate as a function of α_1 (with $\alpha_2 = 1 - \alpha_1$). Figure 5.5, on the other hand, considers a three heterogeneous paths system with the following parameters $\mathcal{F}_1(b) = 0.25\,(b/10)^{0.5} + 0.01$, $\mathcal{F}_2(b) = 0.3\,(b/10)^{0.5} + 0.01$, $\mathcal{F}_3(b) = 0.35\,(b/10)^{0.5} + 0.01$, $\mathcal{B}_1(b) = 15e^{-b/1500} + 0.1$, $\mathcal{B}_2(b) = 10e^{-b/1500} + 0.1$, and $\mathcal{B}_3(b) = 5e^{-b/1500} + 0.1$. (i.e., the first and the last paths have the same characteristics as in the previous example). Figure 5.5 illustrates the contour map of loss rates at various values of $\boldsymbol{\alpha}$. Under the *best* single path streaming approach (i.e., assuming we know which is the best path), we can achieve a loss rate of 11.26%.

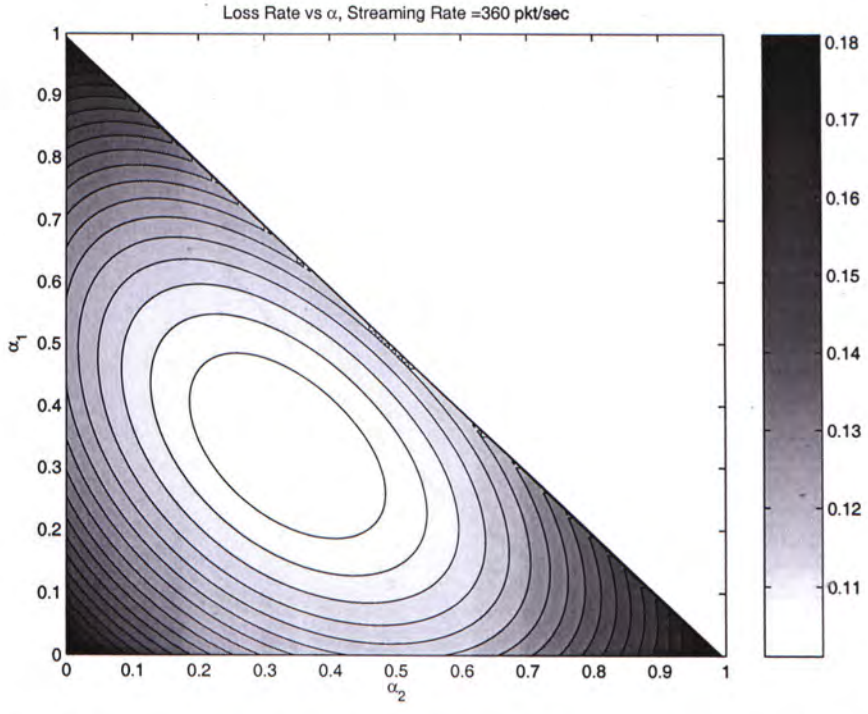


Figure 5.3: Contour map of loss rates under 3 homogeneous paths, with $\boldsymbol{\alpha}^* = [1/3, 1/3, 1/3]$ and corresponding loss rate of 10.1%.

However, under the 2-paths streaming, we reduce the loss rate to 10.7%, with $\boldsymbol{\alpha}^* = [0.902, 0.098]$. We can reduce the loss rate further via the 3-paths streaming approach and achieve a loss rate of 8.89%, with $\boldsymbol{\alpha}^* = [0.661, 0.278, 0.061]$. This example illustrates that the traffic splitting flexibility of the multi-path approach provides us the opportunity to reduce the mean loss rate of a streaming application to a point which would not be possible with a single *best-path* type approach. (As can be noted in the above example, the third path that was added in this example is not the best of the three, yet it allows us to reduce the loss rate further.) It is also important to point out that for this heterogeneous path system, the loss function is also convex and we can easily find a global optimal traffic splitting $\boldsymbol{\alpha}^*$.

The above given examples illustrate the usefulness of the optimization technique. We explore this in more detail in Section 5.3.

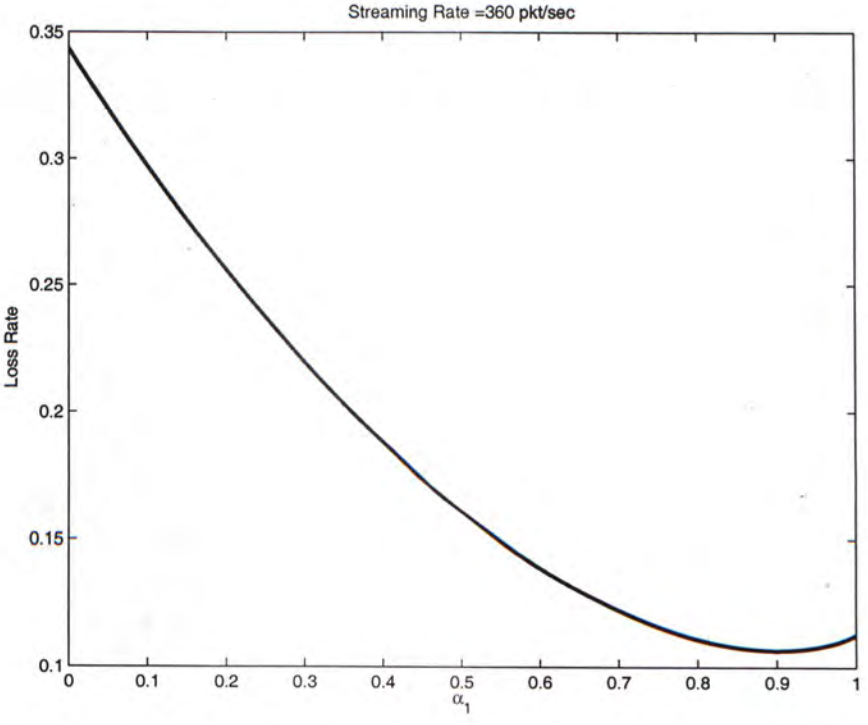


Figure 5.4: Loss rates under 2 heterogeneous paths, with $\boldsymbol{\alpha}^* = [0.902, 0.098]$ and the corresponding loss rate of 10.7%.

5.2.2 Optimization Based on Lag-1 Autocorrelation

As explained in Section 5.1, other metrics (in addition to loss rate) can have a significant effect on the viewing quality of continuous media. One such metric is the lag-1 autocorrelation. Hence, in this section, we consider the problem of finding the right traffic splitting when one wants to optimize the achieved lag-1 autocorrelation function, R_M, rather than the loss rate, P_M. As before, let $\mathcal{F}_j(b)$ denote the functional transition rate from state 0 to state 1 and $\mathcal{B}_j(b)$ denote the functional transition rate from state 1 to state 0 when the streaming traffic on path j is b pkt/sec.

We now consider the case of two-path streaming. As before, $F_j(\alpha_j\lambda) = \frac{\mathcal{F}_j(\alpha_j\lambda)}{\mathcal{F}_j(\alpha_j\lambda)+\mathcal{B}_j(\alpha_j\lambda)}$, for $j = 1, 2$, where we can express the achieved loss rate as

$$P_2 = \alpha_1 F_1(\alpha_1\lambda) + (1-\alpha_1)F_2((1-\alpha_1)\lambda).$$

Let $G_j(\alpha_j\lambda, \delta)$ denote the probability of having two consecutive packet losses

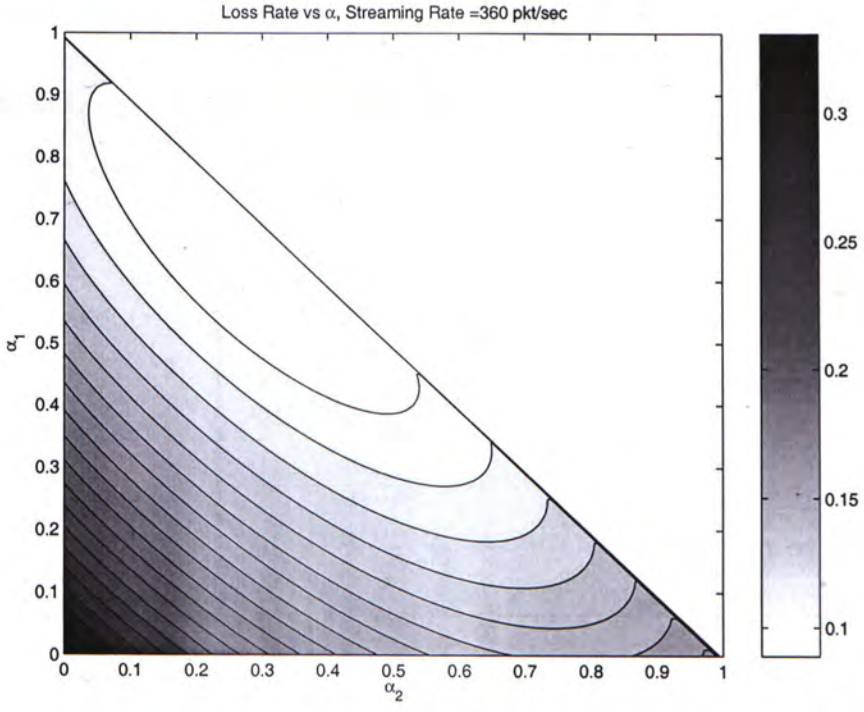


Figure 5.5: Contour map of loss rates under 3 heterogeneous paths, with $\boldsymbol{\alpha}^* = [0.661, 0.278, 0.061]$ and the corresponding loss rate of 8.89%.

at the receiver from path j, for $j = 1, 2$, where these two consecutive packets are separated by δ time units, i.e., $\delta = \frac{1}{\lambda}$. Then,

$$G_j(\alpha_j\lambda, \delta) = \frac{\mathcal{F}_j(\alpha_j\lambda) + \mathcal{B}_j(\alpha_j\lambda)e^{-(\mathcal{F}_j(\alpha_j\lambda)+\mathcal{B}_j(\alpha_j\lambda))\delta}}{\mathcal{F}_j(\alpha_j\lambda) + \mathcal{B}_j(\alpha_j\lambda)}.$$

We can now express the achieved lag-1 autocorrelation function, $R_M[\mathcal{X}_t\mathcal{X}_{t+\delta}]$, which measures the degree of dependency of consecutive packet losses as seen by the receiver, where $\mathcal{X}_t$ is a random variable indicating whether the packet sent at time t is lost or received properly (depending on the state of the Gilbert model) and as before $1/\delta$ is the bandwidth requirement (in units of packets/sec)

of the continuous media streaming application. Specifically,

$$
\begin{aligned}
R_2[\mathcal{X}_t\mathcal{X}_{t+\delta}] &= \frac{E[(\mathcal{X}_t-\overline{\mathcal{X}})(\mathcal{X}_{t+\delta}-\overline{\mathcal{X}})]}{E[(\mathcal{X}_t-\overline{\mathcal{X}})^2]} = \frac{E[\mathcal{X}_t\mathcal{X}_{t+\delta}]-\overline{\mathcal{X}}^2}{E[\mathcal{X}_t^2]-\overline{\mathcal{X}}^2} \\
&= \frac{E[\mathcal{X}_t\mathcal{X}_{t+\delta}]-\overline{\mathcal{X}}^2}{E[\mathcal{X}_t]-\overline{\mathcal{X}}^2} = \frac{E[\mathcal{X}_t\mathcal{X}_{t+\delta}]-\overline{\mathcal{X}}^2}{\overline{\mathcal{X}}-\overline{\mathcal{X}}^2} \\
&= \begin{cases}
\frac{(1-2\alpha_1)F_2((1-\alpha_1)\lambda)G_2((1-\alpha_1)\lambda,\delta)}{P_2-P_2^2} \\
+\frac{2\alpha_1F_1(\alpha_1\lambda)F_2((1-\alpha_1)\lambda)}{P_2-P_2^2} - \frac{P_2^2}{P_2-P_2^2}, \text{when } \alpha_1 < \frac{1}{2}, \\
\frac{F_1(\alpha_1\lambda)F_2((1-\alpha_1)\lambda)-P_2^2}{P_2-P_2^2}, \text{when } \alpha_1 = \frac{1}{2}, \\
\frac{(2\alpha_1-1)F_1(\alpha_1\lambda)G_1(\alpha_1\lambda,\delta)}{P_2-P_2^2} \\
+\frac{2(1-\alpha_1)F_1(\alpha_1\lambda)F_2((1-\alpha_1)\lambda)}{P_2-P_2^2} - \frac{P_2^2}{P_2-P_2^2}, \text{when } \alpha_1 > \frac{1}{2}.
\end{cases}
\end{aligned} \tag{5.8}
$$

We now define the optimal traffic split as a split which minimizes the absolute value of the lag-1 autocorrelation function[1] . Then, to find the optimal traffic split, $\boldsymbol{\alpha}^* = [\alpha_1^*, 1-\alpha_1^*]$, we equate the numerator in Equation (5.8) to zero, i.e.,

$$
\begin{aligned}
&(1-2\alpha_1)F_2((1-\alpha_1)\lambda)G_2((1-\alpha_1)\lambda,\delta) + \\
&\qquad 2\alpha_1F_1(\alpha_1\lambda)F_2((1-\alpha_1)\lambda) - P_2^2 = 0, && \text{when } \alpha_1 < \tfrac{1}{2}, \\
&F_1(\alpha_1\lambda)F_2((1-\alpha_1)\lambda) - P_2^2 = 0, && \text{when } \alpha_1 = \tfrac{1}{2}, \\
&(2\alpha_1-1)F_1(\alpha_1\lambda)G_1(\alpha_1\lambda,\delta) + \\
&\qquad 2(1-\alpha_1)F_1(\alpha_1\lambda)F_2((1-\alpha_1)\lambda) - P_2^2 = 0, && \text{when } \alpha_1 > \tfrac{1}{2}.
\end{aligned}
$$

This equation can be solved using standard numerical methods for finding roots. If more than one solution for α_1 satisfies this equation, then we can check the corresponding loss rate, P_2, to determine which of the possible splitting is more desirable, i.e., we can choose the one with the lowest loss rate.

Note that, when the two paths are homogeneous (i.e., $F_1()$ and $F_2()$ have the same form), then equal splitting of traffic along these two paths (i.e, $\alpha_1 =$

[1]The intuition here is that we are trying to reduce the correlations between packet losses, as explained in Section 5.1.

$\alpha_2 = 1/2$) is a critical point in the above optimization problem. This claim can be easily verified as follows:

$$F_1(\frac{\lambda}{2})F_2(\frac{\lambda}{2}) - [\frac{\lambda}{2}F_1(\frac{\lambda}{2}) + \frac{\lambda}{2}F_2(\frac{\lambda}{2})]^2 = 0.$$

As before, we consider a family of functional transition rates, where the functional Gilbert model for path j, for $j = 1, \ldots, M$, has the following form:

$$\mathcal{F}_j(b) = \beta_j (b/\sigma_j)^{\theta_j} + S_j, \tag{5.9}$$

$$\mathcal{B}_j(b) = \kappa_j e^{-\chi_j b} + \varphi_j \tag{5.10}$$

where $\beta_j, \kappa_j, \chi_j \geq 0$, and $\sigma_j, \theta_j, S_j, \varphi_j > 0$, i.e., $\mathcal{F}_j$ is non-decreasing and $\mathcal{B}_j$ is exponential and decreasing, both functions of the traffic bandwidth b. We then use this family of Gilbert models to illustrate the performance gains of multi-path streaming. Specifically, we consider an application with a bandwidth requirement of 360 pkt/sec (or around 3.84 Mbps). We first consider a system with two homogeneous paths wherein the parameters for $\mathcal{F}(b)$ are $\beta = 0.3$, $\sigma = 10$, $\theta = 0.5$, $S = 0.01$ and the parameters for $\mathcal{B}(b)$ are $\kappa = 10$, $\chi = 1/1500$, $\varphi = 0.1$. Figure 5.6 illustrates the corresponding lag-1 autocorrelation as a function of α_1 (the corresponding α_2 is $1 - \alpha_1$). In this example, the optimized lag-1 autocorrelation equals to zero, with $\boldsymbol{\alpha}^* = [0.5, 0.5]$, with a corresponding loss rate, P_2, of 12.5%.

Now let us consider an application with the same bandwidth requirement of 360 pkt/sec (or around 3.84 Mbps) but under a heterogeneous paths setting. Specifically, we consider a system with two heterogeneous paths wherein $\mathcal{F}_1(b) = 0.25\,(b/10)^{0.5} + 0.01$, $\mathcal{F}_2(b) = 0.35\,(b/10)^{0.5} + 0.01$, $\mathcal{B}_1(b) = 15e^{-b/1500} + 0.1$ and $\mathcal{B}_2(b) = 5e^{-b/1500} + 0.1$. Figure 5.7 illustrates the lag-1 autocorrelation as a function of α_1 (with $\alpha_2 = 1 - \alpha_1$). In this example, we can determine two α_1's (one being smaller than 0.5 and the other being larger than 0.5) which both achieve lag-1 autocorrelation of zero. By comparing the corresponding loss rates, we choose $\boldsymbol{\alpha}^* = [0.550, 0.450]$, at which point the loss

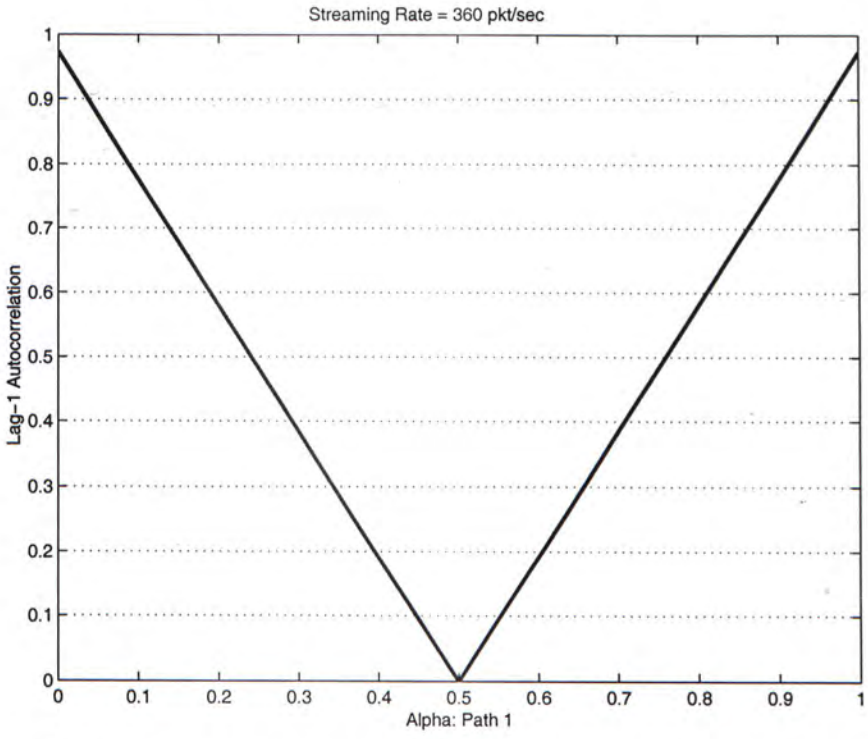


Figure 5.6: Lag-1 autocorrelation under 2 homogeneous paths, with $\boldsymbol{\alpha}^* = [0.5, 0.5]$ and the corresponding loss rate of 12.5%.

rate is approximately 14.9%. (The second optimal lag-1 autocorrelation point corresponds to $\boldsymbol{\alpha}^* = [0.480, 0.520]$ and a loss rate of approximately 16.6%.)

The above given examples illustrate the usefulness of the optimization technique. We explore this in more detail in Section 5.3.

5.3 Experiments

In this section, we consider the system's performance using the performance metrics defined earlier, i.e., packet loss rate, lag-1 autocorrelation of packet losses, and lost packets burst length distribution, all under the two optimization approaches presented in Section 5.2. We consider two types of experiments. In a type A experiment, the servers simply send data packets to the receiver, whereas in a type B experiment, an error erasure code is used to reconstruct some of the lost packets. For all experiments, each data packet has

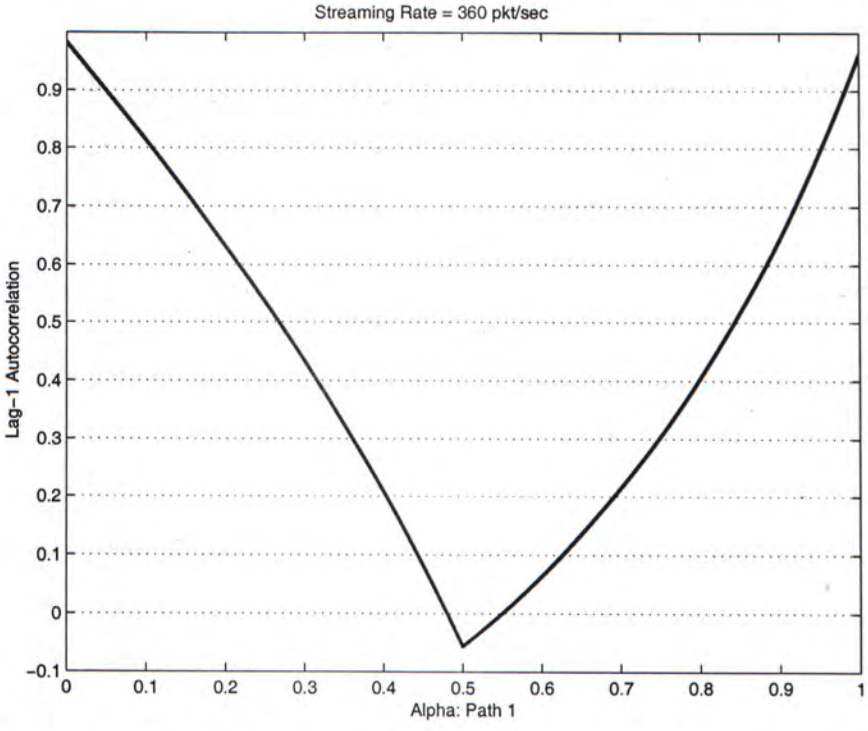


Figure 5.7: Lag-1 autocorrelation under 2 heterogeneous paths, with $\boldsymbol{\alpha}^* = [0.550, 0.450]$ and the corresponding loss rate of 14.9%.

a size of 1400 bytes and we use *sending pattern 3* (refer to Section 7) as our packet quantization method, with a packet group size of $k = 100$. The results presented below are obtained through simulation, using CSIM [18].

5.3.1 Type A Experiment: Without an Erasure Code

In this experiment, we consider two heterogeneous paths with parameters $\mathcal{F}_1(b) = 0.25\,(b/10)^{0.5} + 0.01$, $\mathcal{F}_2(b) = 0.35\,(b/10)^{0.5} + 0.01$, $\mathcal{B}_1(b) = 15e^{-b/1500} + 0.1$ and $\mathcal{B}_2(b) = 5e^{-b/1500} + 0.1$. Tables 5.2 and 5.3 illustrate various performance metrics, such as the optimal splitting vector $\boldsymbol{\alpha}^*$, the achieved loss rate, the achieved lag1-autocorrelation as well as the system's performance when we stream the data using the *best* single path. Figure 5.8 illustrates the corresponding conditional lost packets burst length probability mass function,

Rate (pkt/sec)	Opt. α_1^*	Opt. α_2^*	Opt. Loss Rate	Best SP Loss Rate	Lag1 at Opt. $\boldsymbol{\alpha}^*$
60	0.941	0.059	0.0399	0.0411	0.58507
120	0.933	0.067	0.0571	0.0591	0.65543
360	0.902	0.098	0.1066	0.1126	0.64907

Table 5.2: 2-paths optimization based on loss rate for various sending rates: optimal $\boldsymbol{\alpha}^*$, optimal loss rate, the correspond lag1-autocorrelation, and the loss rate for single best path.

Rate (pkt/sec)	Opt. α_1^*	Opt. α_2^*	Achieved Loss Rate	Optimal Lag1	Best SP Lag 1
60	0.540	0.460	0.0649	0.00377	0.7771
120	0.540	0.460	0.0895	0.00373	0.8834
360	0.550	0.450	0.1864	0.01642	0.9632

Table 5.3: 2-paths optimization based on lag-1 autocorrelation for various sending rates: optimal $\boldsymbol{\alpha}^*$, achieved corresponding loss rate and lag-1 autocorrelation, at optimal $\boldsymbol{\alpha}^*$, and the best single path lag-1 autocorrelation.

conditioned on there being a loss[2] ; this is illustrated for various sending rates and under the two different optimization methods.

We also consider the improvements in various performance measures when we consider an additional path with parameters $\mathcal{F}_3(b) = 0.3\,(b/10)^{0.5} + 0.01$ and $\mathcal{B}_3(b) = 10e^{-b/1500} + 0.1$. Note that this additional path does not possess the best packet loss characteristics. Therefore, this additional path simply adds more path diversity to the data transmission process. Tables 5.4 and 5.5 illustrate the same performance metrics as above, and Figure 5.9 illustrates the corresponding conditional lost packets burst length probability mass function, under 3-paths streaming, for various sending rates and under the two different optimization methods.

[2]We present the probability mass function rather than the probability distribution function, as we believe it depicts the results of the experiment better.

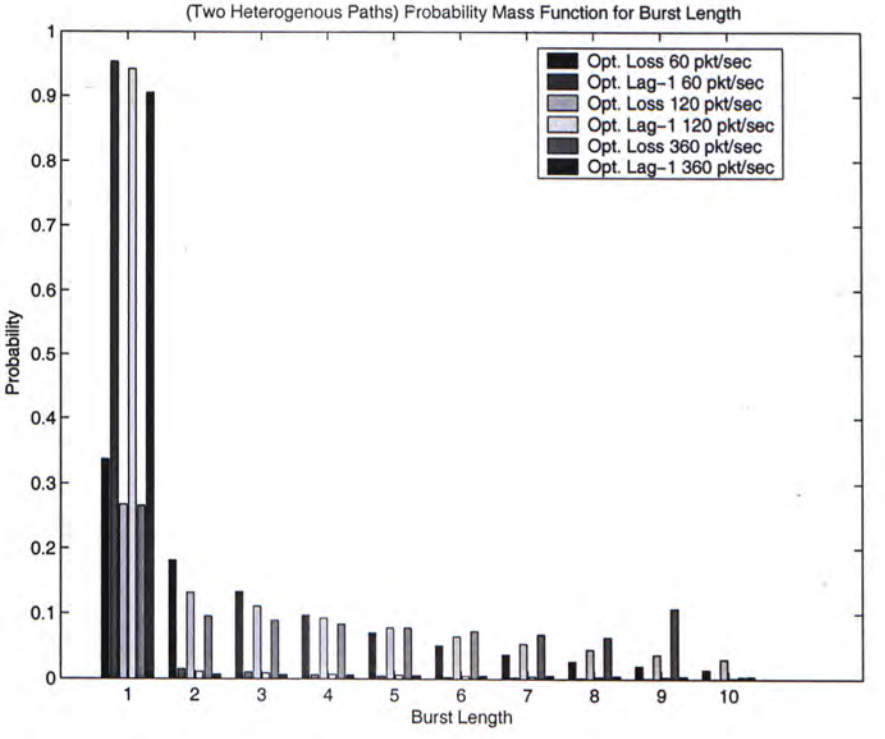


Figure 5.8: Probability mass functions for lost packets burst length for the two optimization methods under 2-paths streaming.

In both experiments, we observe that loss rate-based optimization can significantly reduce the packet loss rate but it has a higher lag-1 autocorrelation than the lag-1 autocorrelation-based optimization method. Also, the lost packet burst length probability mass function under the lag-1 autocorrelation-based optimization method is more skewed toward a single packet loss — this should result in better visual quality of continuous media. When we move from a 2-path system to a 3-paths system, we observe that there is an improvement in terms of loss rate, lag-1 autocorrelation, and lost packet burst length probability mass function, when one uses the loss rate-based optimization method. However, we did not observe an improvement in the lag-1 autocorrelation when using the lag-1 autocorrelation-based optimization method. In general, additional paths reduce the lost packets burst length for both optimization methods. Overall, we observe that both optimization methods result in better system performance (under several metrics) than single path streaming (even

Rate (pkt/sec)	Opt. $\boldsymbol{\alpha}^*$	Opt. loss rate	Best SP loss rate	Lag1 at Opt. $\boldsymbol{\alpha}^*$
60	[0.70,0.04,0.26]	0.0348	0.0411	0.29955
120	[0.71,0.04,0.25]	0.0492	0.0591	0.38131
360	[0.61,0.07,0.32]	0.0883	0.1126	0.23731

Table 5.4: 3-paths optimization based on loss rate for various sending rates: optimal $\boldsymbol{\alpha}^*$, optimal loss rate, corresponding lag-1 autocorrelation, and loss rate for single best path.

Rate (pkt/sec)	Opt. $\boldsymbol{\alpha}^*$	Achieved loss Rate	Optimal Lag1	Best SP Lag 1
60	[0.5, 0.0, 0.5]	0.0400	-0.00389	0.7771
120	[0.5, 0.0, 0.5]	0.0562	-0.00409	0.8838
360	[0.5, 0.0, 0.5]	0.0949	-0.00761	0.9634

Table 5.5: 3-paths optimization based on lag-1 autocorrelation for various sending rates: optimal $\boldsymbol{\alpha}^*$, corresponding achieved loss rate, achieved corresponding lag-1 autocorrelation at optimal $\boldsymbol{\alpha}^*$, and the best single path lag-1 autocorrelation.

when best-path streaming is used).

5.3.2 Type B Experiment: with an Erasure Code

In this experiment, we consider the effects of an erasure code on the various performance measures, such as information loss rate (i.e., packet loss rate after the lost packet reconstruction process), corresponding lag-1 autocorrelation (i.e., after reconstruction), and information burst loss distribution.

Since numerous erasure coding schemes exist [2, 19] we first give a brief explanation of the erasure code used here. We divide a video file into groups of data packets such that each group consists of Ω data packets. Given each group of Ω data packets, we generate $\Upsilon > \Omega$ packets. We refer to these Υ packets as an EC (erasure code) group. The encoding scheme is such that, if the number of lost packets within an EC group is less than or equal to

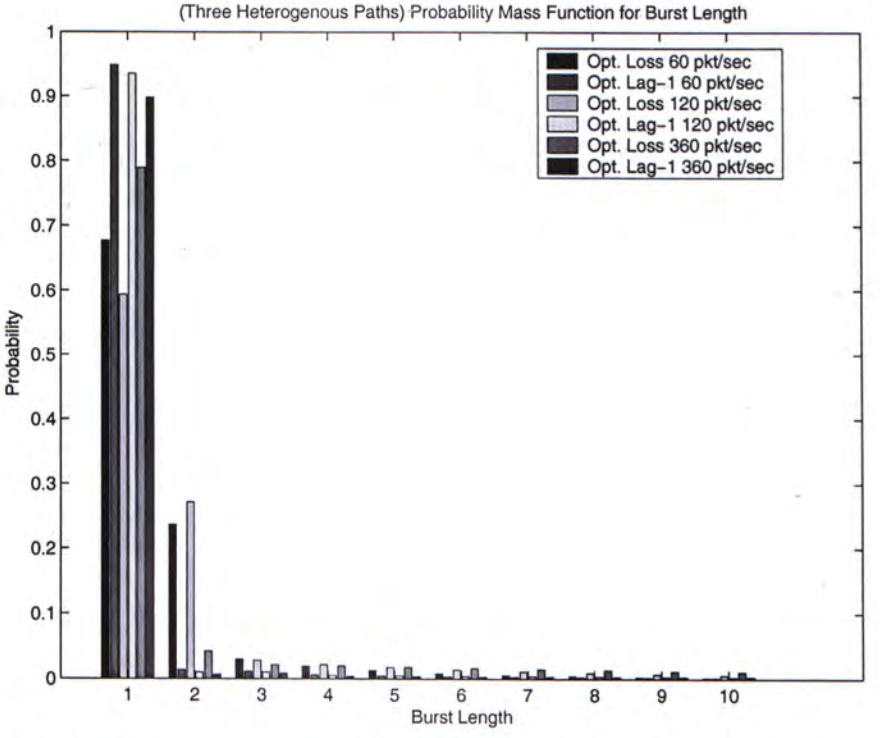


Figure 5.9: Probability mass functions for lost packets burst length for the two optimization methods under 3-paths streaming.

$(\Upsilon - \Omega)$, then we can reconstruct the original Ω data packets within that EC group. In the following experiment, we consider an EC with parameters $\Omega = 32$ and $\Upsilon = 36$. In other words, the bandwidth requirements of the streaming application are increased by 12.5%.

Similarly to the type A experiment, we first consider two heterogeneous paths with parameters $\mathcal{F}_1(b) = 0.25\,(b/10)^{0.5} + 0.01$, $\mathcal{F}_2(b) = 0.35\,(b/10)^{0.5} + 0.01$, $\mathcal{B}_1(b) = 15e^{-b/1500} + 0.1$ and $\mathcal{B}_2(b) = 5e^{-b/1500} + 0.1$. Tables 5.6 and 5.7 illustrate various performance metrics, such as the optimal splitting vector $\boldsymbol{\alpha}^*$, the achieved loss rate, the achieved lag1-autocorrelation as well as the system's performance when we stream the data under the *best* single path.

Figure 5.10 illustrates the corresponding information burst loss probability mass function, for various sending rates and under the two different optimization methods. Again, we consider the improvements in various performance

Rate (pkt/sec)	Opt. α_1^*	Opt. α_2^*	Opt. Inf. loss rate	Best SP Inf. loss rate	Lag-1 at Opt. $\boldsymbol{\alpha}^*$
60	0.87	0.13	0.0279	0.0336	0.67714
120	0.88	0.12	0.0491	0.0573	0.74389
360	0.89	0.11	0.1017	0.1192	0.83736

Table 5.6: 2-paths optimization with erasure code based on loss rate for various sending rates: optimal $\boldsymbol{\alpha}^*$, optimal information loss rate, corresponding lag1-autocorrelation, and information loss rate for single path.

Rate (pkt/sec)	Opt. α_1^*	Opt. α_2^*	Achieved Inf. loss rate	Optimal Lag1	Best SP Lag 1
60	0.50	0.50	0.0577	0.11176	0.85300
120	0.50	0.50	0.0906	0.10033	0.91076
360	0.50	0.50	0.1643	0.07834	0.96706

Table 5.7: 2-paths optimization with erasure code based on lag-1 autocorrelation for various sending rates: optimal $\boldsymbol{\alpha}^*$, achieved information loss rate, lag-1 autocorrelation at optimal $\boldsymbol{\alpha}^*$, and the best single path lag-1 autocorrelation.

measures when we employ an erasure code and an additional path with parameters $\mathcal{F}_3(b) = 0.3\,(b/10)^{0.5} + 0.01$ and $\mathcal{B}_3(b) = 10e^{-b/1500} + 0.1$. Tables 5.8 and 5.9 illustrate the same performance metrics as above, and Figure 5.11 illustrates the corresponding information loss burst probability mass function under 3-paths streaming, for various sending rates and under the two different optimization methods.

From above experiments, we observe that when one employs the optimization method based on loss rate with an erasure code, one can further reduce the information loss rate but the lag-1 autocorrelation does not improve. But when an additional path is available, one can also reduce the information loss rate and the lag-1 autocorrelation at the same time. If one uses the optimization based on lag-1 autocorrelation with an erasure code, the information loss

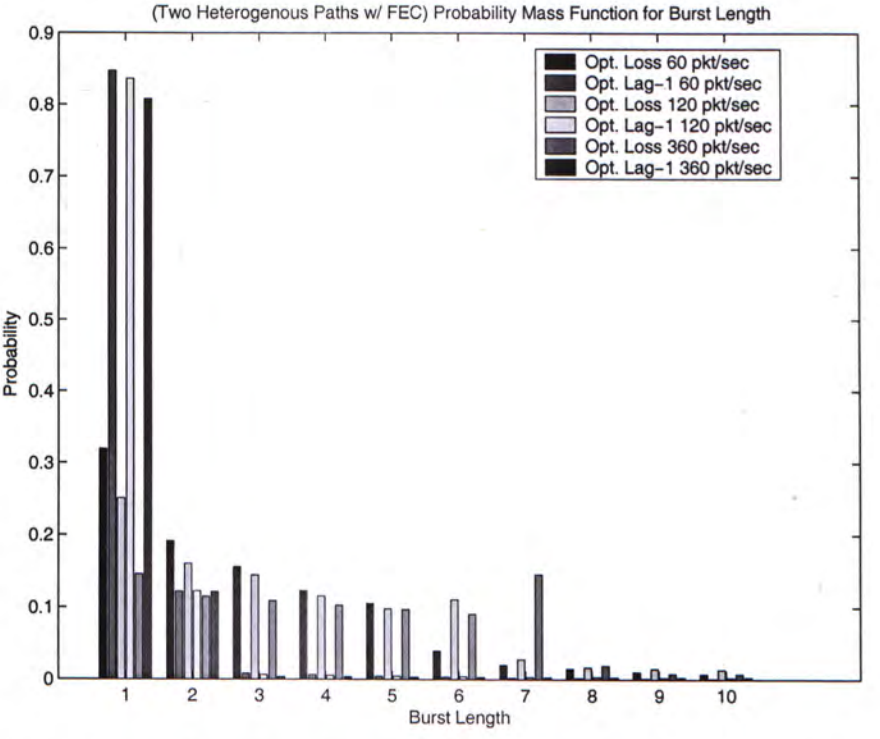


Figure 5.10: Probability mass functions for lost packet burst length for 2 paths with erasure code for the two optimization methods.

rate reduces slightly but the lag-1 autocorrelation actually increases[3] . Again, additional paths can improve the information loss rate and the lag-1 autocorrelation. In general, additional paths reduce the lost packets burst length for both optimization methods. We can also observe that both optimization methods result in significantly better system performance (under several metrics) than single path streaming (even when a best-path approach is used).

[3]This increase is likely due to certain α^* quantization. However, detailed investigation of quantization effects is outside the scope of this thesis.

Rate (pkt/sec)	Opt. $\boldsymbol{\alpha}^*$	Opt. loss rate	Best SP loss rate	Lag1 at Opt. $\boldsymbol{\alpha}^*$
60	[0.58,0.12,0.30]	0.0179	0.0333	0.28914
120	[0.62,0.11,0.27]	0.0365	0.0571	0.33126
360	[0.68,0.11,0.21]	0.0799	0.1199	0.51958

Table 5.8: 3-paths with erasure code and optimization based on loss rate for various sending rates: optimal $\boldsymbol{\alpha}^*$, optimal loss rate, corresponding lag1-autocorrelation, and loss rate for single path.

Rate (pkt/sec)	Opt. $\boldsymbol{\alpha}^*$	Achieved loss Rate	Optimal Lag1	Best SP Lag 1
60	[0.34, 0.33, 0.33]	0.0337	0.03735	0.8525
120	[0.34, 0.33, 0.33]	0.0592	0.02368	0.9103
360	[0.34, 0.33, 0.33]	0.1217	0.00181	0.9672

Table 5.9: 3-paths with erasure code and optimization based on lag-1 autocorrelation for various sending rates: optimal $\boldsymbol{\alpha}^*$, corresponding achieved loss rate, corresponding achieved lag-1 autocorrelation at optimal $\boldsymbol{\alpha}^*$, and the best single path lag-1 autocorrelation.

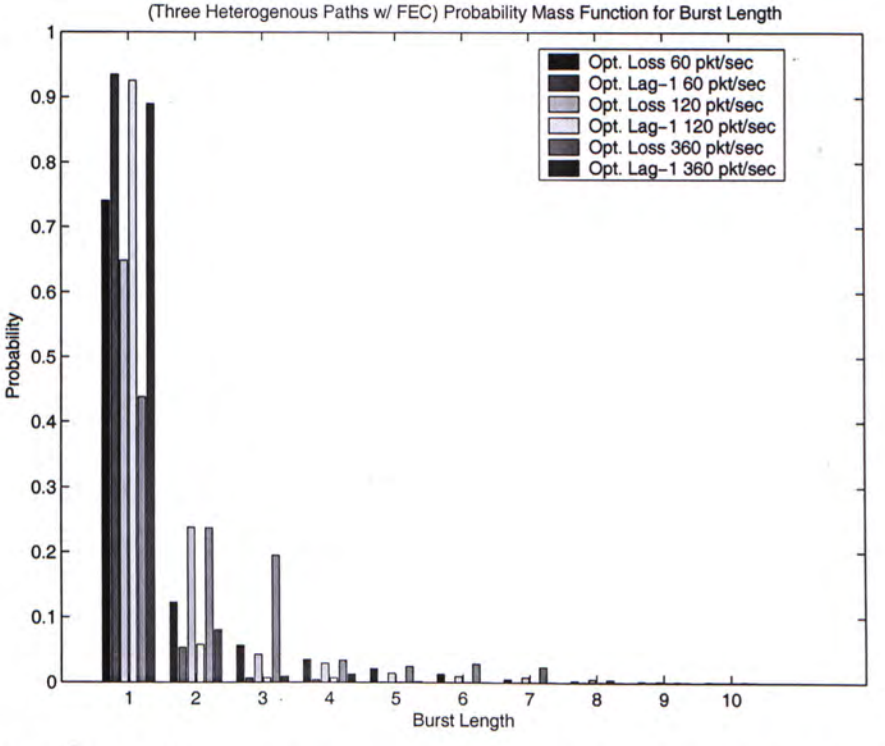


Figure 5.11: Probability mass functions for lost packets burst length under with an erasure code, for the two optimization methods under 3-paths streaming.

Chapter 6

NS Simulations

In this chapter, we evaluate the performance of single path (SP) streaming vs. multiple path (MP) streaming using the NS-2 [3] simulator. NS-2 is a packet level simulator which allows us to study the performance measures (as defined in Section 4.1.1) under more realistic traffic and Internet protocols (such as UDP).

6.1 Simulation Setup

We consider at most three senders (S_1, S_2 and S_3) and one receiver C. Figure 6.1 illustrates our simulation topology. Each sender transmits the video data, at a constant rate, to the receiver C using the UDP protocol, with packet sizes of 1400 bytes. The data traffic goes through two types of links: (1) wide/higher capacity links (represented by solid lines) and (2) narrow/lower capacity links (represented by dotted lines). Each wide link has a bandwidth of 10 Mbps while the bandwidth of a narrow link is 3 Mbps. Each link has a different propagation delay and the propagation delay is generated using an exponential random variable with a mean of 200 ms. The streaming application has a sending rate of 1.5 Mbps which consumes 50% of the bandwidth of a narrow link. The actual sending rate of each sender is a function of the traffic load distribution. Unless stated otherwise, an equal distribution is used, e.g.,

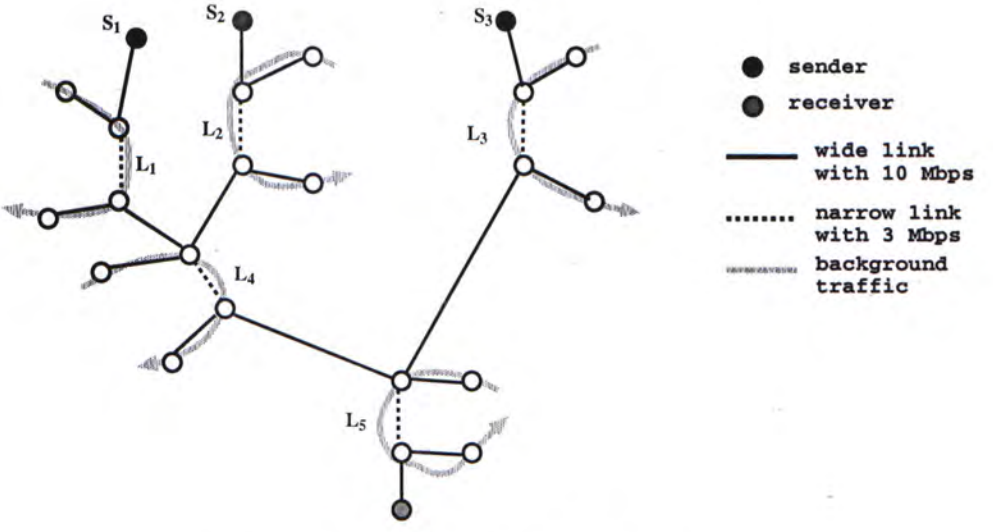


Figure 6.1: Simulation topology.

for MP streaming with three senders (sending data in a round-robin manner), the sending rate of each sender is 0.5 Mbps. Background traffic (represented by grey arrows) is introduced at different narrow links. The background traffic is generated using exponential on/off sources. The average "on" time plus the average "off" time of these on/off sources is equal to 1 second. During the "on" times, the background source generates UDP traffic with a constant rate of 3 Mbps, which can saturate the capacity of the traversed narrow links. In the following experiments we vary the amount of "on" time within an average of 1 second period. For example, a background traffic rate of 1.8 Mbps represents an average "on" time of 0.6 seconds for an average of 1 second on/off period. There are three possible sets of background traffic locations. One set of local background traffic occurs on the narrow links L_i where $i = 1, 2, 3$. This background traffic competes with the corresponding sender S_i $(i = 1, 2, 3)$ for the bandwidth resources of the narrow links L_1, L_2, and L_3, respectively. The second set of background traffic occurs on the narrow link L_4. This background traffic competes with senders S_1 *and* S_2 for the bandwidth resource of the narrow link L_4. The third set of background traffic occurs on the narrow link L_5. This background traffic competes with *all* three senders for the bandwidth

resource of the narrow link L_5. Unless stated otherwise, SP streaming is done from sender 1 and dual-path streaming is done from senders 1 and 3.

6.2 Simulation Result

Experiment 1 (Data Loss Rate): Figure 6.2 illustrates the data loss rates

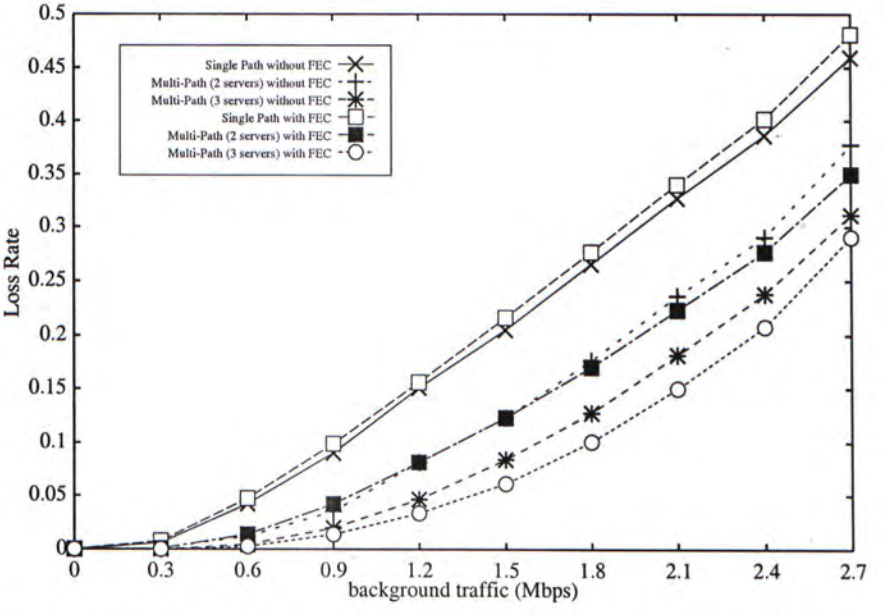


Figure 6.2: Loss rate with FEC parameters $n = 10$ and $k = 8$.

for SP and MP streaming. In this simulation, we vary the average background traffic through the narrow links L_1, L_2, and L_3 from 0 Mbps to 2.7 Mbps. (Note that the senders do not share points of congestion in this case.) From this figure, we observe the following. Firstly, MP streaming can achieve a significant reduction in the data loss rate as compared to SP streaming. Secondly, employment of FEC may actually increase the data packet loss rate; for example, the data loss rate of SP streaming with FEC is a bit higher than the data loss rate of SP without FEC. Thirdly, the improvements in the data loss rate achieved through the use of MP streaming *without* FEC is higher than that achieved through the use of FEC by adding it to SP streaming. This is potentially due to the fact that the use of FEC (with SP streaming) introduces

additional traffic into the (already) congested network and hence results in higher data losses. On the other hand, the use of MP streaming achieves a significant reduction in data loss rate without introduction of additional network traffic.

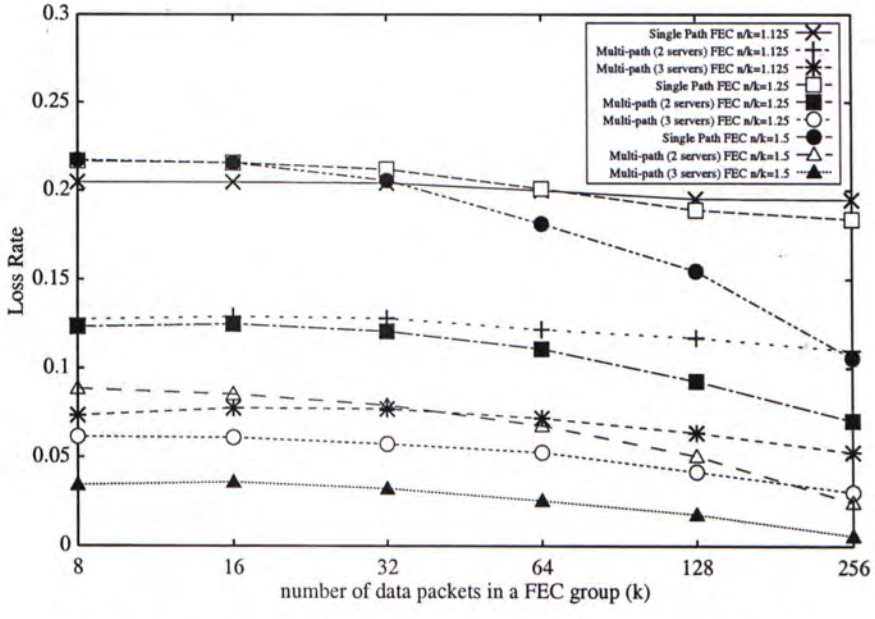


Figure 6.3: Loss rate as a function of n/k ratio and k

Experiment 2 (Data Loss Rate as a function of FEC parameters): In this experiment, we study the effects of FEC parameters on the data loss rate. Again, we vary the FEC parameters as in Section 4.2. Figure 6.3 illustrates the data loss rate when a background traffic of 1.5 Mbps is used on each of the narrow links L_1, L_2, and L_3. We observe that:

- Increasing the degree of redundancy under SP streaming may not necessarily reduce the data loss rate, one reason being that introducing additional traffic (due to higher degree of redundancy) into an already congested network may result in higher packet loss rates. Hence, MP streaming may have a higher chance of decreasing the data loss rate with higher degrees of redundancy, i.e., with less traffic being introduced per path.

- MP streaming can significantly reduce data loss rate as compared to SP streaming.

In summary, we observe that increasing the amount of redundancy (by increasing the n/k ratio) or increasing the FEC group size (and hence potentially suffering higher latency at the receiver with a need for larger buffer sizes) may not result in significant reduction in data loss rate, for either SP or MP streaming. On the other hand, taking advantage of multiple independent paths, can reduce the data loss rate significantly.

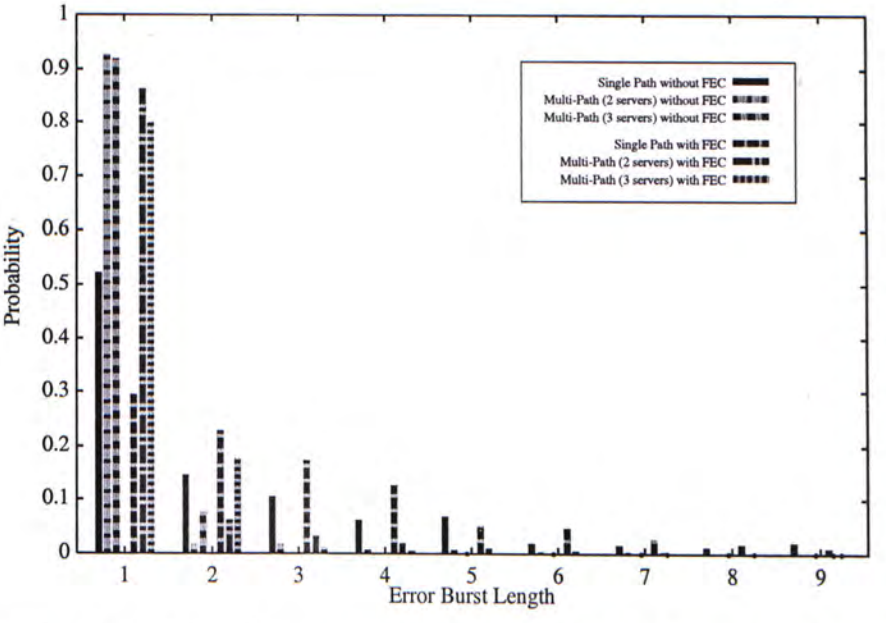


Figure 6.4: Conditional probability mass function for error burst length.

Experiment 3 (Conditional Error Burst Length): In this experiment, we compare the conditional burst length distribution, conditioned on there being at least one loss, of the SP and MP approaches. In this case a background traffic of 2.4 Mbps is used on each of the narrow links L_1, L_2 and L_3. The conditional probability mass function[1] of error burst length is given in Figure 6.4, where we observe that MP streaming has a stochastically smaller data packet burst length than SP streaming.

[1] As in Section 4.2 we illustrate the probability mass function rather than the probability distribution function, as we believe it depicts the results of the experiment better.

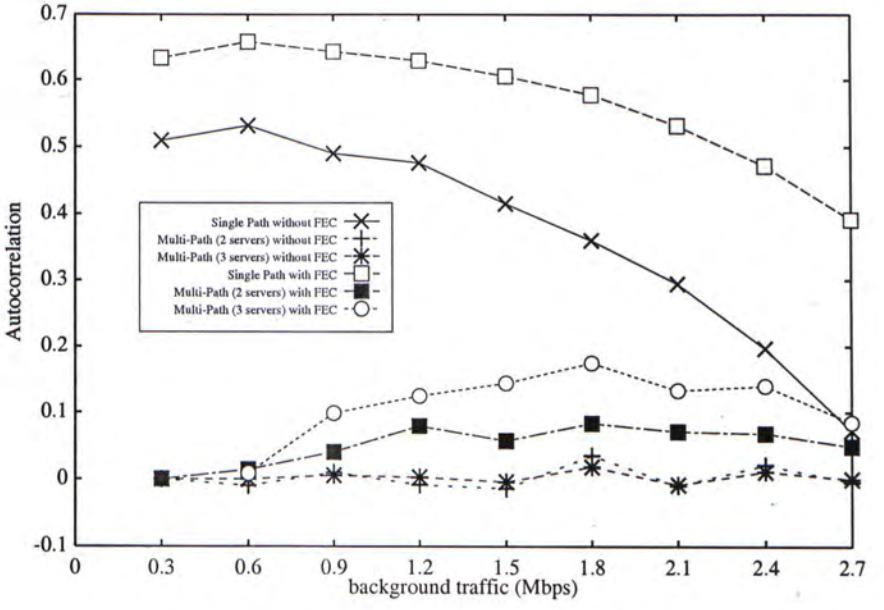


Figure 6.5: Lag-1 autocorrelation.

Experiment 4 (Lag-1 Autocorrelation): In this experiment, we study lag-1 autocorrelation of packet losses for both SP and MP streaming. Figure 6.5 illustrates the lag-1 autocorrelation as we vary the background traffic on the narrow links L_1, L_2 and L_3. We observe the following.

- Without use of FEC, the MP lag-1 autocorrelation is close to zero (as derived in Section 4.1.1), i.e., the losses appear nearly uncorrelated when streaming over multiple independent paths. On the other hand, the correlation of losses with SP streaming can be quite high.

- With use of FEC, lag-1 autocorrelation may increase. We believe that a similar explanation (as given in Experiment 4 of Section 4.2) holds here. However, we still observe that the MP lag-1 autocorrelation is significantly lower than the SP lag-1 autocorrelation (under the same FEC scheme).

Lastly, the decrease in lag-1 autocorrelation as a function of higher background traffic may be counter-intuitive. One explanation may be that the "no losses" (i.e., the packets that are received successfully) in the resulting stream tend to

be more "random" as congestion on the network increases.

Experiment 5 (Effects of Load Distribution among Senders): In previous experiments, all senders transmitted packets in a round-robin manner and hence the load on all senders (i.e., the amount of data streamed from each sender) was the same. In this experiment, we study the effects of different load distributions on the resulting loss characteristics observed at the receiver. Specifically, we consider the following configurations.

Configuration 1 Streaming from sender 1 only.

Configuration 2 Equal distribution of load between senders 1 and 3 only.

Configuration 3 Equal distribution of load among all senders.

Configuration 4 Sender 1 streams 1/6 of the data, sender 2 streams 1/6 of the data, and sender 3 streams 2/3 of the data.

Figure 6.6 depicts the data loss rate and the lag-1 autocorrelation of these configurations. In this experiment, equal distribution of load (configuration 3) tends to achieve a lower data loss rate and lag-1 autocorrelation.

Experiment 6 (Sensitivity Analysis): In this experiment, we study the relative performance of MP streaming vs. SP streaming when the SP streaming is performed over the *best* of the available paths (please refer to Section 4.2 for a more detailed explanation of "best path" streaming and the motivation for making this comparison). Specifically, we consider a two senders system with only senders S_1 and S_3 transmitting packets. The background traffic on L_1 is fixed at 1.5 Mbps, and the background traffic on L_3 is varied from 0.3 to 2.7 Mbps. In this scenario, the best-path approach believes (based on collected measurements) that the path originating at sender S_3 experiences the least losses. Therefore, the best-path streaming approach always uses the path originating from sender S_3. We also consider a very *simple* MP streaming approach, which streams the data in a round-robin manner from S_1 and S_3.

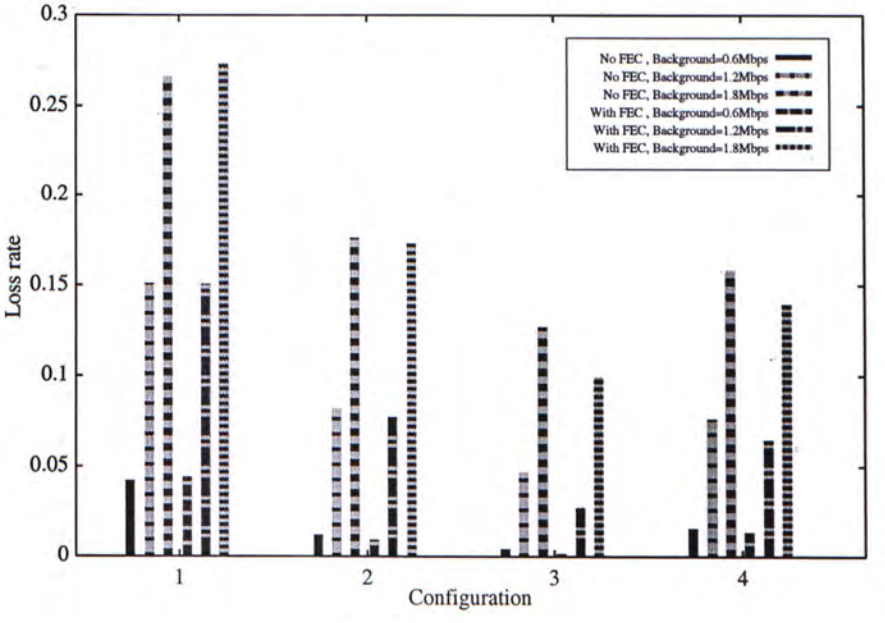
Figure 6.7 illustrates the *relative* loss rate (using several different FEC schemes), which is defined as the data loss rate of dual-path streaming divided by the data loss rate of best-path streaming. Hence, a relative loss rate of less than 1, implies that simple dual-path streaming is more robust as compared to best-path streaming. As in Section 4.2, we observe that simple dual-path (round-robin) streaming does quite well compared to best-path streaming, even when there is significant differences in loss characteristics between the two paths. Of course, in cases where the best path has much better loss characteristics and with relatively little redundant information, the best-path approach has a lower data loss rate. Hence, we believe that the MP approach is more robust as compared to best-path streaming.

Experiment 7 (Effects of Shared Points of Congestion on Various Performance Metrics): In this experiment, we study the effects of *shared points-of-congestion*, between the paths used by the different senders, on various performance measures. Here, the background traffic is sent through the narrow links L_3 and L_4. Note that, having background traffic on L_4 implies that senders 1 and 2 share the same point-of-congestion. Again, we consider the four configurations described in Experiment 5 above.

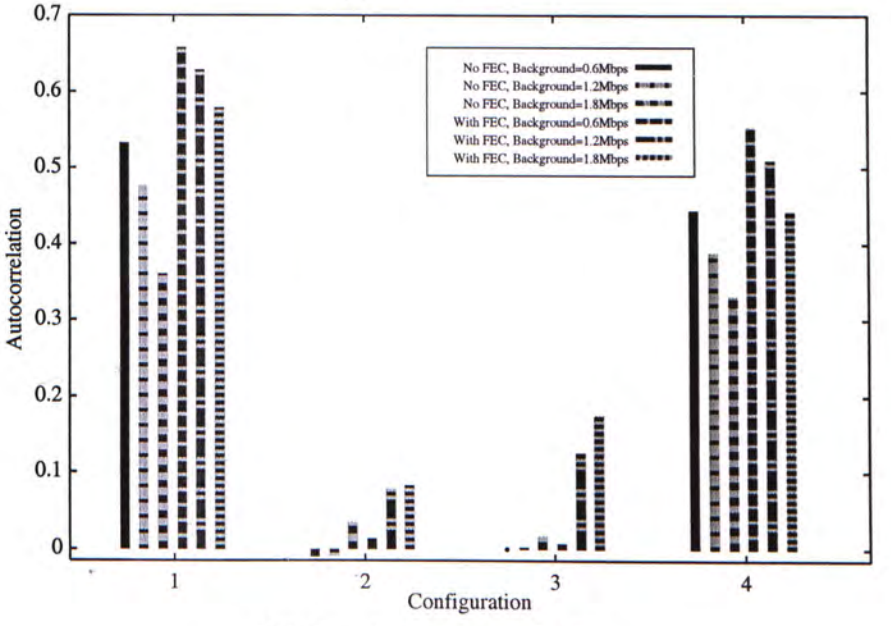
Figure 6.8 illustrates the data loss rate and lag-1 autocorrelation for these configurations, when FEC is used, with $(n = 10, k = 8)$. Moreover, we vary the background traffic on the two narrow links L_3 and L_4 among the following values: 0.6 Mbps, 1.2 Mbps, 1.8 Mbps, and 2.4 Mbps. From this figure, we observe the following.

- MP streaming (configurations 2, 3, 4) has a lower data loss rate as compared to SP streaming (configuration 1).
- Detecting shared points of congestion is important, as including a greater number of paths/senders (under such conditions) in the transmission may adversely affect the data loss rate.

- Shared points of congestion adversely affect the lag-1 autocorrelation metric. For example, configuration 3 has a higher lag-1 autocorrelation than configuration 2.



(a) Loss Rate



(b) Lag-1 Autocorrelation

Figure 6.6: Loss rate and Lag-1 autocorrelation under different load distributions

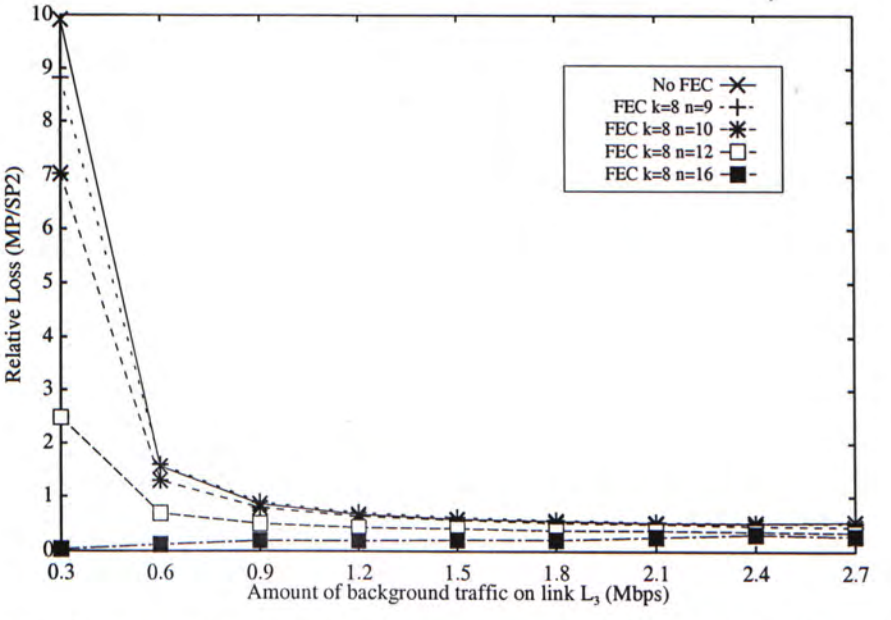
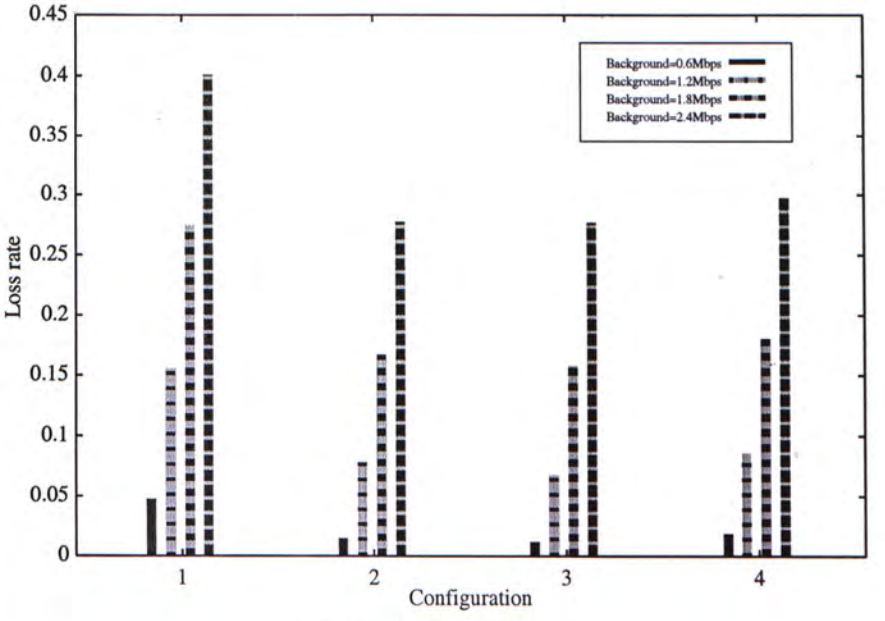
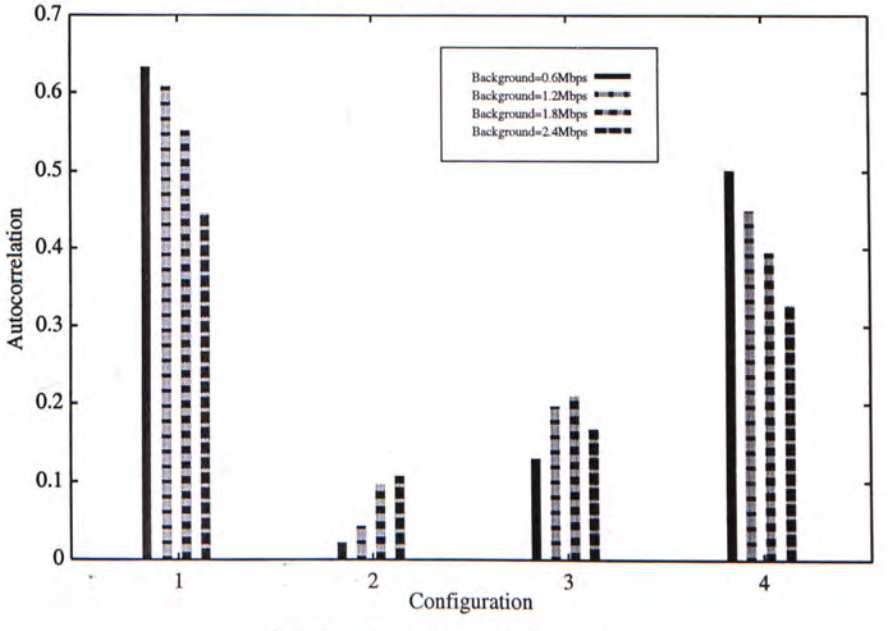


Figure 6.7: Relative loss rate when background traffic on link L_3 and FEC group size are varied.



(a) Data Loss Rate



(b) Lag1-autocorrelation

Figure 6.8: Effects of shared points-of-congestion on data loss rate and lag-1 autocorrelation with FEC ($n = 10, k = 8$).

Chapter 7

Quantization of Traffic Splitting Vector

In Chapter 5 we presented an approach to determining an optimal traffic splitting vector $\boldsymbol{\alpha}^*$, based on optimization of a particular performance metric, such as the lag-1 autocorrelation function and the mean loss rate. However, the traffic splitting vector $\boldsymbol{\alpha}$ only provides us the relative desired traffic loading ratios among the M servers/paths. To realize multi-path streaming, we need to map a given traffic splitting vector $\boldsymbol{\alpha}$ to a packet sending pattern, where different sending patterns can have a significant effect on the performance metrics considered earlier (as described below). We refer to this as the *quantization* process.

Assume that a streaming application needs to transmit N packets to a receiver. Define $\mathcal{P}_N$ as a row vector of dimension N whose j^{th} element equals $\mathcal{P}_N(j) \in \{1, 2, \ldots, M\}$. In other words, when $\mathcal{P}_N(j) = k$, the k^{th} server is responsible for sending the j^{th} packet of the streaming application. Then, given a traffic splitting vector $\boldsymbol{\alpha}$, one simple way to perform the quantization

process is as follows[1] :

$$P_N(j) = \begin{cases} 1 & \text{if } 1 \le j \le N\alpha_i, \\ 2 & \text{if } N\alpha_1 + 1 \le j \le N(\alpha_1 + \alpha_2), \\ 3 & \text{if } N(\alpha_1 + \alpha_2) + 1 \le j \le N\sum_{l=1}^{3} \alpha_l, \\ \vdots & \vdots \\ M & \text{if } N(\sum_{l=1}^{M-1}) + 1 \le j \le N\sum_{l=1}^{M} \alpha_l. \end{cases}$$

When N is large, this quantization procedure can closely approximate the optimal splitting vector $\boldsymbol{\alpha}^*$. However, there are some inherent problems with using this approach. These include

- *Higher lag-1 autocorrelation.* Since we potentially assign a large number of consecutive packets to a path, congestion on that path may result in a high lag-1 autocorrelation. As described in Appendix A, high positive lag-1 autocorrelation can significantly affect the visual quality of a continuous media stream; it can also reduce the effectiveness of an error correction scheme.

- *Larger burst length.* Again, sending more consecutive packets on the same path may increase the probability of having longer bursts of lost packets. This could also result in poor visual quality of a continuous media stream. Likewise, it can also have detrimental effects on the number of losses that an erasure scheme will be able to correct.

- *Greater difficulty in adapting to network conditions.* Whenever there is a change in network conditions, one may opt to adapt to this change and switch some traffic from one path to another path (i.e., recompute $\boldsymbol{\alpha}^*$). The above quantization process essentially commits all N packets to specific paths and hence makes adaptation and its implementation more difficult.

[1]For simplicity of presentation, we consider $\alpha_i N$, for all i, to be an integer. During implementation, some form of rounding is necessary.

To overcome these problems, we propose the following approach. We consider a *packet group* of size k packets, where $k << N$ (e.g., $k = 128$ or 256). We then divide the N application packets into N/k groups. Within a packet group, we distribute packets according to the traffic splitting vector $\boldsymbol{\alpha}$, while trying to disperse as much as possible packets belonging to the same path. As long as the network conditions do not change, the sending pattern repeats for each sending group. On the other hand, if there is any change in network conditions, the receiver can react to these changes by computing a new splitting vector $\boldsymbol{\alpha}$ and signaling all M servers with the new sending pattern.

Note that different sending patterns within a packet group can affect various performance metrics (as observed by the receiver) such as the lag1 autocorrelation, the burst length of lost packets, and to some degree, the achieved loss rate. In other words, the sending pattern can affect not only the visual quality of continuous media, but also the effectiveness of error correction schemes.

We consider three sending patterns.

- *Sending Pattern 1:* Assume that the packet group size is k. Assign packets from $k \sum_{j=1}^{i-1} \alpha_j$ to $k \sum_{j=1}^{i} \alpha_j$ to the i^{th} server/path. For example, if $k = 10$ and $\boldsymbol{\alpha} = [0.1, 0.3, 0.6]$, the sending pattern will be:

  1222333333 1222333333 1222333333 ...

  In this case, the average length of consecutive packets from the same server/path is equal to $1 * (1/10) + 3 * (3/10) + 6 * (6/10) = 4.6$. Clearly, in this case a large number of consecutive packets still comes from the same server/path. Hence, using this sending pattern can result in adverse effects on both, the lag-1 autocorrelation and the burst length of lost packets.

- *Sending Pattern 2:* This is a pseudo round robin method wherein consecutive packets are assigned to different servers/paths as much as possible. When a server depletes all its packets within a packet group (according

to the computed $\boldsymbol{\alpha}$), the other servers continue to fill in the sending pattern, again, in a round robin manner. For example, if $k = 10$ and $\boldsymbol{\alpha} = [0.1, 0.3, 0.6]$, the sending pattern will be:

1232323333 1232323333 1232323333 ...

In this case, the average length of consecutive packets from the same server/path is equal to $1 * (6/10) + 4 * (4/10) = 2.2$. This method tries to reduce the correlation between consecutive packet losses at the beginning of the packet sending group. However, if the α_is are not equal, then towards the end of a packet group there maybe a significant number of consecutive packets from the same server/path, which may introduce a long burst of lost packets.

- *Sending Pattern 3:* In this method, we space out the packets from each server as evenly as possible within a packet group. Specifically, to determine the sending pattern, we process the servers/paths, in rounds, from the smallest α to the largest α. In each round, the server with the current smallest α is responsible for transmitting $k\alpha$ packets. These packets are spread out such that the spacing is equal to the number of un-assigned packets within the packet sending group divided by $k\alpha$. A more precise description of this algorithm is given in Appendix B. For example, if $k = 10$ and $\boldsymbol{\alpha} = [0.1, 0.3, 0.6]$, the sending pattern using this method will be

  1233233233 1233233233 1233233233 ...

  In this case, the average length of consecutive packets from the same server/path is equal to $1 * (4/10) + 2 * (6/10) = 1.6$. This method tries to reduce the burst length of lost packets.

Pseudo code for producing Sending Pattern 3:

Input: α, M;

Output: Sending pattern in pkt group sending_pattern[]

```
for (i = 1 to M) {/* For each server i */
/* Find the number of pkt for server i within */
/* a pkt sending group (group size = k) */
    pkt_in_group[i] = k * α_i;
}
Sort the M servers according to pkt_in_group[],
                        from smallest to largest;
/* For each slot s in the sending group */
for (s = 1 to k) {
    /* Initialize all slots in pkt group to EMPTY */
    sending_pattern[s] = EMPTY ;
}
/* Initialize number of empty slot */
remaining_slot = k;

for (i = 1 to M) { /* Process each server one by one */
    /* Process only if this server has pkt to send */
    if (pkt_in_group[i] > 0) {
        packet_processed = 0;
        required_spacing = floor (remaining_slot / pkt_in_group[i]);
        skip_space = required_spacing;
        /* For each slot s in the pkt sending group */
        for (s = 1 to k){
            if (packet_processed < pkt_in_group[i]){
            /* If server i still has pkt to send */
                if (sending_pattern[s] == EMPTY) {
                /* If slot is not yet assigned */
```

```
            if (skip_space != required_spacing) { /* skip slot */
                skip_space++;
            } else { /* Assign pkt to slot */
                    /* Assign path i to slot s */
                    sending_pattern[s] = i; skip_space=1;
                    packet_processed ++;
                } /* else termination */
            } /* if termination */
        } /* if termination */
    } /* for loop termination */
    remaining_slot -= pkt_in_group[i];
} /* if termination */
} /* for loop termination */
```

Chapter 8

Prototype Implementation and Experiments

Following the approach proposed in Chapter 3, we implemented a deployable multi-path multimedia streaming system prototype. Using this prototype system, we carry out experiments to study the performance of using different traffic splitting vector. We consider the three performance metrics as mentioned in Chapter 3 as well as the actual visual output.

8.1 Multi-path Streaming Prototype

The prototype system is divided in into two layers, they are i) the lower multi-path streaming module layer and ii) the upper application layer. Given the multi-path streaming parameters (i.e., media file, number of servers, sending pattern, FEC parameters), the multi-path streaming module performs the basic streaming functions such as reading data from media source file, packetizing the data, processing FEC, transmitting packets, measuring statistic, and decoding video. These processes are performed in an multi-path manner and with dynamic adaptation capability (i.e., sending pattern and FEC parameters can be changed throughout the streaming process). The upper application layer is responsible to carry out the communications between the receiver and senders,

as well as to design the streaming parameters. This division of functionalities allows other applications to use our multi-path streaming module easily by following the API of the module. For instance, our researcher collaborator in the USA is integrating the streaming module into a peer-to-peer file sharing system.

In the remaining part of this chapter, experiments with this prototype system are discussed.

8.2 Experiments

In this section, experiments on the multi-path streaming system are carried by using the functional Gilbert model as described in Chapter 5 as the packet drop model. We stream a MPEG1 file which requires a playback rate of 174.5 kbps (around 170 packet/sec with packet size 1000 byte). FEC with $k = 64$, $n = 72$ is used. This increases the packet sending rate to around 192 packet/sec. Two senders, whose path to the receiver carry different loss characteristics, are used in this experiment. For each path, the only source of packet loss is caused by the function Gilbert model.The functions for the FGM are:

FGM functions for Path 1 (with better loss characteristics):

$$\mathcal{F}_j(b) = 10 * \frac{b}{150}, \tag{8.1}$$

$$\mathcal{B}_j(b) = 190 * \frac{150}{b}. \tag{8.2}$$

FGM functions for Path 2 (with worse loss characteristics):

$$\mathcal{F}_j(b) = 60 * \frac{b}{150}, \tag{8.3}$$

$$\mathcal{B}_j(b) = 140 * \frac{150}{b}, \tag{8.4}$$

where b equals to 192 packet/sec in this experiment.

Using the approach as proposed in Chapter 5, if we optimize for the the loss rate, the optimal traffic splitting is $[0.741, 0.259]$.

Four cases with different traffic splitting vector are studied:

1. Single path with *better* loss characteristics,
2. Single path with *worse* loss characteristics,
3. Dual path using *even* traffic splitting,
4. Dual path using *optimal* traffic splitting.

Packet losses statistics are measured at the receiver throughout the whole streaming process. Video frames are transcoded to JPEG files to allow visual quality comparison.

Test case	loss rate before FEC	lag-1 auto-corr before FEC	avg burst length before FEC	loss rate after FEC
1.	6.48%	0.399	1.781	1.57%
2.	35.48%	0.360	2.423	35.53%
3.	6.86%	-0.042	1.030	0.89%
4.	3.42%	0.191	1.290	0.07%

Table 8.1: Prototype experiments: Average loss statistics under different traffic splitting.

Table 8.1 shows the average statistic measured for each test cases. For the statistics before FEC, we measure the statistic of the resultant packet receiving sequence after merging packets from different paths. Figure 8.1 and Figure 8.2 show the sequences of video frames extracted from the four test cases at the same video time with respect to the start of the video.

From the above results, we observe that:

1. Different loss rate (after FEC) affects the visual quality: When we relate the after FEC loss rate in Table 8.1 with the video output in Figure 8.1 and Figure 8.2, we find that higher loss rate (after the FEC operation) gives poorer video quality. When the loss rate is extremely high, for example 35.53% in Case 2, the video is totally ruined and viewing quality is totally unacceptable. When the loss rate is improved to 1.57% as in Case 1, some damaged frames can be found in some segment of the video sequence. If some important data is lost (e.g., I frame in the MPEG1 standard), scenes mixing may also occurred, as shown in the captured video output. When the loss rate is further improved to 0.89% in Case 3, video distortions happen less frequently and most of the distortions are some "blocking effects" as shown in the video frame extracted. When the loss rate is very low, e.g. 0.07% in Case 4, distortions are very rare and most of them are unnoticeable by human eye.

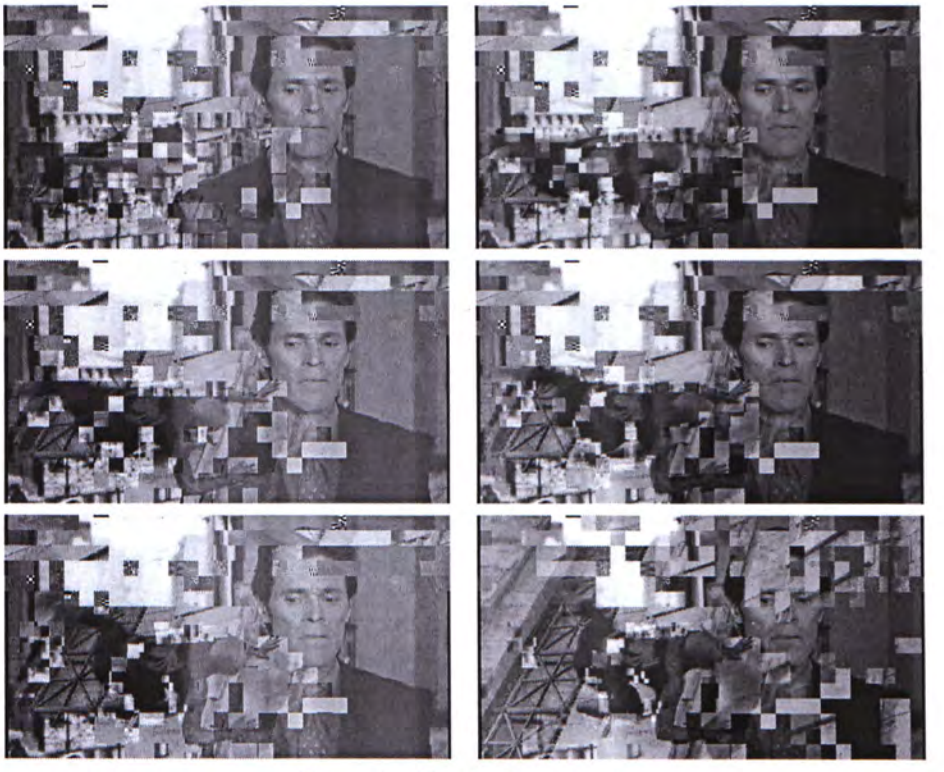
2. Introducing FEC may cause adverse effect: Improper adding FEC may not improve the resultant information loss rate. In Case 2, we find that loss rate before the FEC operation is less than the loss rate after the FEC operation. It means that the FEC performs poorly in this case. Adding FEC increases the loading to a congested path which may worsen the loss characteristics of the paths. It results in an adverse effect that more packets are loss.

3. FEC performs better under the multiple paths streaming setting: As suggested in Chapter 4, multiple paths can reduce the average lag-1 auto-correlation and statistically shorten the error burst length. This enhances the error correction capability of FEC. The loss rate before FEC for Case 1 and Case 3 are 6.48% and 6.86% respectively, in this case, using the best path in Case 1 can gives a lower loss rate before the FEC process. However, after processing the FEC, it becomes 1.57%

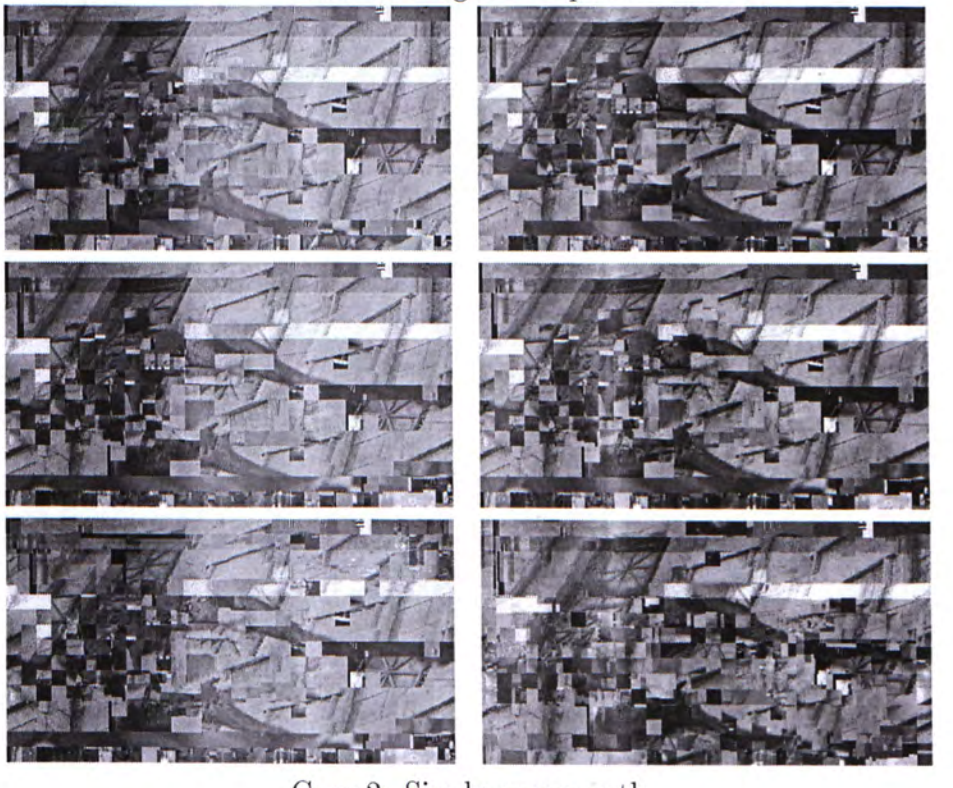
for Case 1 and 0.89% for Case 3. The visual quality, which is strongly related the after FEC loss rate, is much better in Case 3. It shows that simply using the best single path for video streaming may not be a good choice.

4. A path with a higher loss characteristics may still be worth to use: Although path 2 is a bad path, it can still be used to share a fraction of the workload to improve the resultant quality. From the statistic table and video output, the two multiple paths test cases give better performance than the two single best path test cases.

5. Using the optimal traffic splitting vector can achieve better performance: When we assign traffic loading according to the optimized traffic splitting vector on multiple paths, the loss rate is the lowest in both cases, that are, before FEC and after FEC. It shows that simply splitting the traffic evenly on each paths may not get the most benefits from using multiple paths(although it gives packet loss with least correlation).

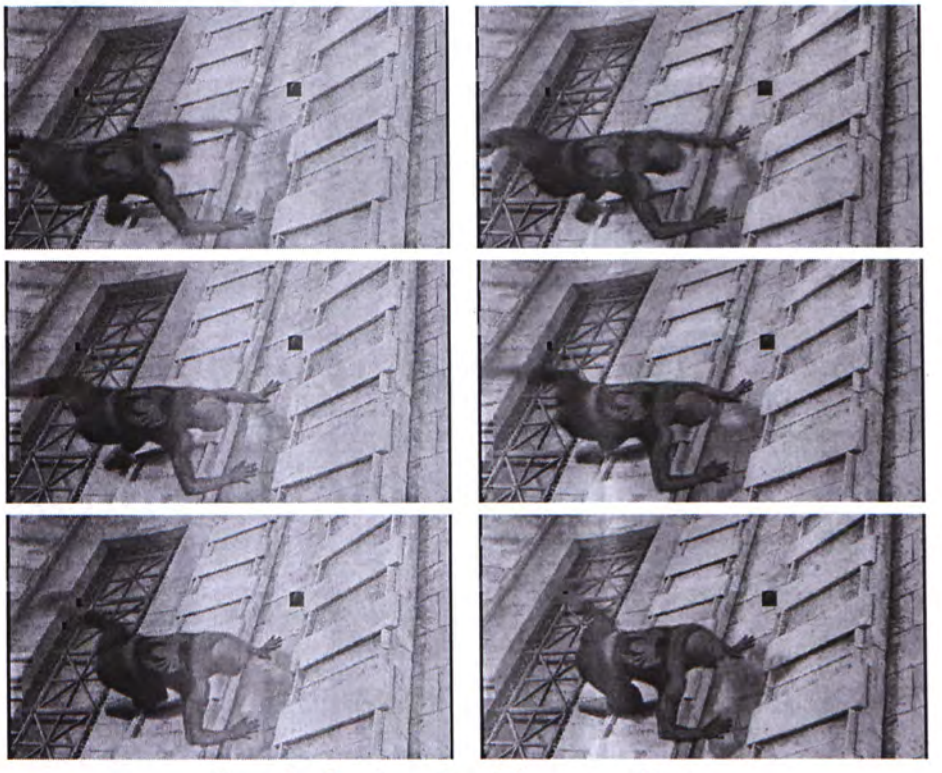


Case 1: Single best path



Case 2: Single worse path

Figure 8.1: Prototype experiments: Video frames output.



Case 3: Dual path with even splitting

Case 4: Dual path with optimal splitting

Figure 8.2: Prototype experiments: Video frames output.

Chapter 9

Other Design Issues and Considerations

In this chapter, some design issues and considerations in using our multi-path streaming approach are discussed. Requirements and overheads are mentioned. A specific technical issue, detecting Share Point of Congestion, is discussed in more details.

9.1 Requirements and Overheads

- Data dispersion overhead: The same piece of media data is needed to be fully replicated in multiple senders to allow multi-path streaming. This might increase the network loading and slow down the delivery of new content. However, our approach can work closely with the rapidly growing peer-to-peer file sharing systems, in which media data is already widely spread across the entire network. Our collaborator in US is integrating our streaming module into a peer-to-peer file sharing system.

- Higher playback delay penalty: Different paths exhibit different delay, The path with the longest delay latency becomes the bottleneck of the video decoding process. The receiver has to wait for the data from the "slowest" path before further processing the data. Thus, the receiver

needs to implement a larger buffer in order to absorb the delay differences between multiple paths. This leads to a longer startup latency as well as a larger memory demand. In practical implementation, this delay penalty and extra memory requirement are in fact not very significant. For a MPEG1 file to be decoded smoothly, most decoders require a buffer of up to three or more seconds. Comparing with the round trip time (RTT) in the current Internet, which is in the order of milliseconds, the differences of delay between paths is relatively small.

Another source of delay comes from the FEC process, the receiver has to buffer the entire FEC group before processing it. Consider a MPEG1 file which has a bandwidth requirement of 1.3 to 1.4 Mbps. If the data packet size is 1400 byte. the packet sending rate will be around 120 packet/sec. If we use $k = 120$ for FEC, the receiver has to inject an additional delay of one second in order to collect the whole FEC group. As discussed earlier, larger FEC group size can correct more errors with a tradeoff of longer delay. On the contrary, due to the less correlated packet loss in multi-path streaming, a smaller FEC group size can be used in multi-path streaming to achieve a better FEC correcting capability.

- More complicated protocol design and implementation: In order to allow multi-path streaming together with dynamic adaption ability. The underlining protocol should be deigned in a robust and efficient manner. Together with the statistic measurement requirement, the program coding in real implementation is more complex than most normal streaming application. We have already successfully implemented a robust multi-path streaming prototype. Other applications can easily plug-in our streaming module to receive the benefits from multi-path streaming.

- Synchronization between the senders and receiver: Communication channels between the receiver and each senders should be established to exchange necessary information. All senders should agree on the same streaming setting (e.g., number of senders, sending pattern, FEC value) before the startup of a stream and before each adaption. If either sender uses a different setting, overlapped sending of data, or missing of data, or data with mismatch FEC setting may happen. Overlapped data sending wastes system and network resource. It may also lead to worsening of the loss characteristics of channels as discussed in Chapter 5. Missing of data will definitely result in a poorer viewing quality. Receiving data with different FEC settings may waste those FEC redundant packets. In the worst case, if the implementation and protocols are not well designed, it may even mislead the FEC decoder to produce some garbage data, which may seriously harm the media decoding process and result in a poor viewing quality.

- Network measurement and optimization complexity: Path characteristics of each path should be known in order to assign the traffic splitting vector accurately. Specifically, the functions for the FGM for each path should be "known" in order to carry out the optimization process. This is still an ongoing future work of our research. Moreover, the optimization process increases the computational overhead in the receiver. Another workaround is to use some heuristic to shift the workload from some "poorer paths", which exhibit higher loss rate, to some "better paths". In both methods, path quality for each path should be measured throughout the whole streaming session because the characteristics of each paths may vary from time to time. Moreover, detection of share point of congestion (will be discussed in the next section) is also required.

9.2 Share Point of Congestion (SPOC)

Special care on SPOC should be taken during adaption and traffic splitting vector assignment. If two paths share the same point of congestion, shifting load among these paths would not give any benefit. For example in Experiment 7 of Chapter 6. Shifting load from S1 and S3 to S2 (from configuration 2 to configuration 3) would adversely increase the lost rate. This is because L4 is a SPOC between S1 and S2. Thus, it is important to detect whether two paths share the same congestion point.

Rubenstein [20] suggested a method to detect whether two flows (paths) contain a SPOC by end point measurement. The basic idea is measuring the correlation of adjacent probe packets between two flows (paths) - cross-measures M_x as well as measuring the correlation of adjacent probe packets from the same flow (path) - auto-measures M_a. If $M_x > M_a$, then two flows (paths) share the same POC. On the other hand, if $M_x < M_a$, then two flows (paths) do not share a POC. Two methods to compute M_a and M_x are proposed. One bases on packet losses and the other bases on packets delay. The probe packets can be either in-band (piggyback into the streaming packet) or out-band.

In-band packets are a trivial choice as it can save network bandwidth. However, as the scheme measures M_x of adjacent packets between two paths, two paths sending probe packets at different rate (which happens when different paths stream at different rate in multi-path streaming) lead to an incorrect decision. Consider the case that path 1 sends 75% and path 2 sends 25% of the data. The packet spacing between consecutive packets within path 2 is three time larger than that of path 1. Given packet i from path 1 and packet j from path 2 are adjacent, it is most likely that packet $i + 1$ from path 1 and packet $j + 1$ from path 2 will *not* be adjacent due to the large difference in sending interval. Another weakness of in-band packets is that some paths

in the system may carry 0% loading, however, we still want to detect if these paths contains SPOC with other paths. Thus, in our scheme, out-band probe is necessary to ensure probe packets are sent on all paths at the same rate.

One way to reduce the traffic of sending out-band packets is to perform piggybacking whenever it is possible. That is, for a path with responsible fraction greater than zero, we try to use in-band packets if possible. It can be implemented as following. When it is time to send a probe packet, it will check if a data packet is going to be sent. If there is, tags the data packet as probe packet, else sends a out-of-band probe packet explicitly.

Chapter 10

Conclusion

In this thesis, we address the problem of providing QoS guarantees for streaming pre-stored continuous media using an application-layer multi-path streaming approach. An advantage of this approach, as compared to approaches that require support of lower layers and resource reservation schemes, is that the complexity of QoS provision and guarantee can be pushed to the network edge. It improves the scalability and deployability of a streaming applications and at the same time, provide a certain level of QoS guarantees. Using the conventional Gilbert model, our results indicate that in general, multi-path streaming exhibits better loss characteristics than single-path streaming (with or without use of an erasure code), which should result in a higher viewing quality of the received continuous media.

We further extend our work by considering the functional Gilbert model, which is more versatile in capturing the dependency of an application's sending rate and the loss characteristics of a path. We show that under the functional Gilbert model representation, any valid traffic splitting will have a lower packet loss rate than a single path streaming approach. We then focus on two optimization approaches, one which tries to minimize the received loss rate and another which optimizes the lag-1 autocorrelation. In general, we observe that optimization based on loss rate can achieve a lower packet loss rate but has a

higher lag-1 autocorrelation and lost packets burst length than the optimization based on lag-1 autocorrelation. When we increase the path diversity (i.e., include more paths in continuous media streaming), we can lower the packet loss rate, lag-1 autocorrelation and lost packet burst length, if one uses the loss rate optimization approach. For lag-1 autocorrelation-based optimization, path diversity will only improve on packet loss rate.

When we add an erasure code to the application (i.e., it also implies that we increase the bandwidth requirements of an application), one can further reduce the information loss rate, lag-1 autocorrelation, and lost packets burst length, if one uses the loss rate optimization approach. For lag-1 autocorrelation-based optimization, employing an erasure code reduces slightly the information loss rate. Path diversity with an erasure code can generally improve all three performance metrics for both optimization methods.

We also build a multi-path streaming system prototype by following our proposed multi-path streaming approach. Experiments are carried out on the prototype system which further validate our multi-path streaming method. This prototype system also enables other applications to easily receive the benefits of multi-path streaming.

Bibliography

[1] D. L. Gall, Communications of the ACM (April 1991).

[2] R. E. Blahut, *Theory and Practice of Error Control Codes*, Addison Wesley, January 1983.

[3] http://www.isi.edu/nsnam/ns/, *The Network Simulator - ns-2.*

[4] N. F. Maxemchuk, Dispersity routing, in *the IEEE International Conference on Communications*, San Francisco, California, June 1975.

[5] N. F. Maxemchuk, Dispersity routing in high-speed networks, in *the Workshop on Very High Speed Networks*, Greenbelt, Maryland, March 1991.

[6] N. F. Maxemchuk, Dispersity routing in an atm network, in *the IEEE INFOCOM*, San Francisco, California, March/April 1993.

[7] H. Chu and K. Nahrstedt, Dynamic multi-path communication for video traffic, in *the Hawiian International Conference on System Science*, Hawaii, January, 1997.

[8] Y. J. Liang, E. G. Steinbach, and B. Girod, Multi-stream voice over ip using packet path diversity, in *the IEEE Fourth Workshop on Multimedia Signal Processing*, Cannes, France, October 2001.

[9] Y. J. Liang, E. G. Steinbach, and B. Girod, Real-time voice communication over the internet using packet path diversity, in *the ACM Multimedia Conference*, Ottawa, Canada, September/October 2001.

[10] J. Apostolopoulos, Reliable video communication over lossy packet networks using multiple state encoding and path diversity, in *the Visual Communications and Image Processing*, San Jose, California, January 2001.

[11] J. Apostolopoulos, T. Wong, W. Tan, and S. Wee, On multiple description streaming with content delivery networks, in *the IEEE Infocom*, New York, New York, June 2002.

[12] T. Nguyen and A. Zakhor, Distributed video streaming over internet, in *the SPIE Conference on Multimedia Computing and Networking*, San Jose, California, January 2002.

[13] T. Nguyen and A. Zakhor, Distributed video streaming with forward error correction, in *the International Packetvideo Workshop*, Pittsburg, Pennsylvania, April 2002.

[14] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison Wesley, 1993.

[15] J.-C. Bolot, S. Fosse-Parisis, and D. Towsley, Adaptive FEC-Based Error Control for Internet Telephony, in *INFOCOM*, 1999.

[16] P. Morse, *Queues, Inventories, and Maintenance*, John Wiley, 1958.

[17] S. Ross, *Stochastic Processes*, John Wiley, 1996.

[18] http://www.mesquite.com/, *CSIM18.*

[19] A. McAuley, Reliable broadband communication using a burst erasure correcting code, in *Proc. ACM SIGCOMM '90; (Special Issue Computer Communication Review)*, 1990.

[20] D. Rubenstein, J. Kurose, and D. Towsley, Detecting shared congestion of flows via end-to-end measurement, in *the ACM Sigmetrics Conference*, Santa Clara, California, June, 2002.